Analysing and Comparing Encodability Criteria for Process Calculi
(Technical Report)

Kirstin Peters
TU Dresden, Germany

Rob van Glabbeek
NICTA†, Sydney, Australia
Computer Science and Engineering, UNSW, Sydney, Australia

August 05, 2015

Abstract

Encodings or the proof of their absence are the main way to compare process calculi. To analyse the quality of encodings and to rule out trivial or meaningless encodings, they are augmented with quality criteria. There exists a bunch of different criteria and different variants of criteria in order to reason in different settings. This leads to incomparable results. Moreover it is not always clear whether the criteria used to obtain a result in a particular setting do indeed fit to this setting. We show how to formally reason about and compare encodability criteria by mapping them on requirements on a relation between source and target terms that is induced by the encoding function. In particular we analyse the common criteria full abstraction, operational correspondence, divergence reflection, success sensitiveness, and respect of barbs; e.g. we analyse the exact nature of the simulation relation (coupled simulation versus bisimulation) that is induced by different variants of operational correspondence. This way we reduce the problem of analysing or comparing encodability criteria to the better understood problem of comparing relations on processes.

In the following we present the Isabelle implementation of the underlying theory as well as all proofs of the results presented in the paper Analysing and Comparing Encodability Criteria as submitted to EXPRESS/SOS’15.

---

*Supported by funding of the Excellence Initiative by the German Federal and State Governments (Institutional Strategy, measure ‘support the best’).

†NICTA is funded by the Australian Government through the Department of Communications and the Australian Research Council through the ICT Centre of Excellence Program.
# Contents

1 Relations 3
   1.1 Basic Conditions .................................................. 3
   1.2 Preservation, Reflection, and Respection of Predicates .......... 6

2 Process Calculi 11
   2.1 Reduction Semantics ............................................... 11
       2.1.1 Observables or Barbs ....................................... 13

3 Simulation Relations 17
   3.1 Simulation .......................................................... 17
   3.2 Contrasimulation ................................................... 22
   3.3 Coupled Simulation ............................................... 24
   3.4 Correspondence Simulation ....................................... 25
   3.5 Bisimulation ....................................................... 30
   3.6 Step Closure of Relations ....................................... 40

4 Encodings 46

5 Relation between Source and Target Terms 54
   5.1 Relations Induced by the Encoding Function ..................... 54
   5.2 Relations Induced by the Encoding and a Relation on Target Terms 77
   5.3 Relations Induced by the Encoding and Relations on Source Terms and Target Terms 120

6 Success Sensitiveness and Barbs 151

7 Divergence Reflection 155

8 Operational Correspondence 156
   8.1 Trivial Operational Correspondence Results .................... 157
   8.2 (Strong) Operational Completeness vs (Strong) Simulation .... 158
   8.3 Weak Operational Soundness vs Contrasimulation ............... 167
   8.4 (Strong) Operational Soundness vs (Strong) Simulation ....... 167
   8.5 Weak Operational Correspondence vs Correspondence Similarity 175
   8.6 (Strong) Operational Correspondence vs (Strong) Bisimilarity .. 181

9 Full Abstraction 201
   9.1 Trivial Full Abstraction Results ................................ 201
   9.2 Fully Abstract Encodings ....................................... 203
   9.3 Full Abstraction w.r.t. Preorders ................................ 210
   9.4 Full Abstraction w.r.t. Equivalences ............................ 217
   9.5 Full Abstraction without Relating Translations to their Source Terms 227

10 Combining Criteria 235
   10.1 Divergence Reflection and Success Sensitiveness ............. 237
   10.2 Adding Operational Correspondence ............................ 238
   10.3 Full Abstraction and Operational Correspondence ............ 271
1 Relations

1.1 Basic Conditions

We recall the standard definitions for reflexivity, symmetry, transitivity, preorders, equivalence, and inverse relations.

**abbreviation** preorder Rel ≡ preorder-on UNIV Rel

**abbreviation** equivalence Rel ≡ equiv UNIV Rel

A symmetric preorder is an equivalence.

**lemma** symm-preorder-is-equivalence:

```plaintext
fixes Rel :: ('a × 'a) set
assumes preorder Rel
and sym Rel
shows equivalence Rel
using assms unfolding preorder-on-def equiv-def
by simp
```

The symmetric closure of a relation is the union of this relation and its inverse.

**definition** symcl :: ('a × 'a) set ⇒ ('a × 'a) set where

symcl Rel = Rel ∪ Rel −1

For all (a, b) in R, the symmetric closure of R contains (a, b) as well as (b, a).

**lemma** elem-of-symcl:

```plaintext
fixes Rel :: ('a × 'a) set
and a b :: 'a
assumes elem: (a, b) ∈ Rel
shows (a, b) ∈ symcl Rel
and (b, a) ∈ symcl Rel
by (auto simp add: elem symcl-def)
```

The symmetric closure of a relation is symmetric.

**lemma** sym-symcl:

```plaintext
fixes Rel :: ('a × 'a) set
shows sym (symcl Rel)
by (simp add: symcl-def sym-Un-converse)
```

The reflexive and symmetric closure of a relation is equal to its symmetric and reflexive closure.

**lemma** refl-symm-closure-is-symm-refl-closure:

```plaintext
fixes Rel :: ('a × 'a) set
shows symcl (Rel −1) = (symcl Rel)−1
by (auto simp add: symcl-def refl)
```

The symmetric closure of a reflexive relation is reflexive.

**lemma** refl-symcl-of-refl-rel:

```plaintext
fixes Rel :: ('a × 'a) set
and A :: 'a set
assumes refl-on A Rel
shows refl-on A (symcl Rel)
using assms
by (auto simp add: refl-on-def symcl-def)
```

Accordingly, the reflexive, symmetric, and transitive closure of a relation is equal to its symmetric, reflexive, and transitive closure.
The reflexive closure of a symmetric relation is symmetric.

The reflexive closure of a reflexive relation is the relation itself.

The symmetric closure of a symmetric relation is the relation itself.

The reflexive and transitive closure of a preorder Rel is Rel.

The reflexive and transitive closure of a relation is a subset of its reflexive, symmetric, and transitive closure.

If a preorder Rel satisfies the following two conditions, then its symmetric closure is transitive: (1) If \((a, b)\) and \((c, b)\) in Rel but not \((a, c)\) in Rel, then \((b, a)\) in Rel or \((b, c)\) in Rel. (2) If \((a, b)\) and \((a, c)\) in Rel but not \((b, c)\) in Rel, then \((b, a)\) in Rel or \((c, a)\) in Rel.
lemma \( \text{symm-closure-of-preorder-is-trans} \):

fixes \( \text{Rel} \) :: \( ('a 	imes 'a) \text{ set} \)

assumes \( \text{condA} \): \( \forall a \ b \ c. \ (a, b) \in \text{Rel} \land (c, b) \in \text{Rel} \land (a, c) \notin \text{Rel} \) \( \rightarrow \) \( (b, a) \in \text{Rel} \lor (b, c) \in \text{Rel} \)

and \( \text{condB} \): \( \forall a \ b \ c. \ (a, b) \in \text{Rel} \land (a, c) \in \text{Rel} \land (b, c) \notin \text{Rel} \) \( \rightarrow \) \( (b, a) \in \text{Rel} \lor (c, a) \in \text{Rel} \)

and \( \text{reflR} \): \( \text{refl} \text{Rel} \)

and \( \text{transR} \): \( \text{trans} \text{Rel} \)

shows \( \text{trans} \) \( \text{symcl Rel} \)

unfolding \( \text{trans-def} \)

proof clarify

fix \( a \ b \ c \)

have \( \{[(a, b) \in \text{Rel}; (b, c) \in \text{Rel}] \implies (a, c) \in \text{symcl Rel} \) proof

assume \( (a, b) \in \text{Rel} \land (b, c) \in \text{Rel} \)

with \( \text{transR} \) have \( (a, c) \in \text{Rel} \)

unfolding \( \text{trans-def} \)

by blast

thus \( (a, c) \in \text{symcl Rel} \)

by \( \text{(simp add: symcl-def)} \)

qed

moreover have \( \{[(a, b) \in \text{Rel}; (c, b) \in \text{Rel}; (a, c) \notin \text{Rel}] \implies (a, c) \in \text{symcl Rel} \) proof

assume \( \text{A1} \): \( (a, b) \in \text{Rel} \land \text{A2} \): \( (c, b) \in \text{Rel} \land (a, c) \notin \text{Rel} \)

with \( \text{condA} \) have \( (b, a) \in \text{Rel} \lor (b, c) \in \text{Rel} \)

by blast

thus \( (a, c) \in \text{symcl Rel} \)

proof auto

assume \( (b, a) \in \text{Rel} \)

with \( \text{A2 tranR} \) have \( (c, a) \in \text{Rel} \)

unfolding \( \text{trans-def} \)

by blast

thus \( (a, c) \in \text{symcl Rel} \)

by \( \text{(simp add: symcl-def)} \)

next

assume \( (b, c) \in \text{Rel} \)

with \( \text{A1 tranR} \) have \( (a, c) \in \text{Rel} \)

unfolding \( \text{trans-def} \)

by blast

thus \( (a, c) \in \text{symcl Rel} \)

by \( \text{(simp add: symcl-def)} \)

qed

qed

moreover have \( \{[(b, a) \in \text{Rel}; (b, c) \in \text{Rel}; (a, c) \notin \text{Rel}] \implies (a, c) \in \text{symcl Rel} \) proof

assume \( \text{B1} \): \( (b, a) \in \text{Rel} \land \text{B2} \): \( (b, c) \in \text{Rel} \land (a, c) \notin \text{Rel} \)

with \( \text{condB} \) have \( (a, b) \in \text{Rel} \lor (c, b) \in \text{Rel} \)

by blast

thus \( (a, c) \in \text{symcl Rel} \)

proof auto

assume \( (a, b) \in \text{Rel} \)

with \( \text{B2 tranR} \) have \( (a, c) \in \text{Rel} \)

unfolding \( \text{trans-def} \)

by blast

thus \( (a, c) \in \text{symcl Rel} \)

by \( \text{(simp add: symcl-def)} \)

next

assume \( (c, b) \in \text{Rel} \)

with \( \text{B1 tranR} \) have \( (c, a) \in \text{Rel} \)

unfolding \( \text{trans-def} \)

by blast

by blast
thus \((a, c) \in \text{symcl Rel}\)
by (simp add: symcl-def)
qed
qed
moreover have \([[(b, a) \in \text{Rel}; (c, b) \in \text{Rel}] \implies (a, c) \in \text{symcl Rel}]
proof –
assume \((c, b) \in \text{Rel} \text{ and } (b, a) \in \text{Rel}\)
with tranR have \((c, a) \in \text{Rel}\)
unfolding trans-def
by blast
thus \((a, c) \in \text{symcl Rel}\)
by (simp add: symcl-def)
qed
moreover assume \((a, b) \in \text{symcl Rel} \text{ and } (b, c) \in \text{symcl Rel}\)
ultimately show \((a, c) \in \text{symcl Rel}\)
by (auto simp add: symcl-def)
qed

1.2 Preservation, Reflection, and Respection of Predicates

A relation \(R\) preserves some predicate \(P\) if \(P(a)\) implies \(P(b)\) for all \((a, b)\) in \(R\).

abbreviation rel-preserves-pred :: \(((\alpha \times \alpha) \set \Rightarrow (\alpha \Rightarrow \text{bool}) \Rightarrow \text{bool})\) where
rel-preserves-pred \(\alpha\) \(\beta\) \(\epsilon\) \(\eta\) \(\zeta\) \(\eta\)

abbreviation rel-preserves-binary-pred :: \(((\alpha \times \alpha) \set \Rightarrow (\alpha \Rightarrow \text{bool}) \Rightarrow \text{bool})\) where
rel-preserves-binary-pred \(\alpha\) \(\beta\) \(\epsilon\) \(\eta\) \(\zeta\) \(\eta\)

A relation \(R\) reflects some predicate \(P\) if \(P(b)\) implies \(P(a)\) for all \((a, b)\) in \(R\).

abbreviation rel-reflects-pred :: \(((\alpha \times \alpha) \set \Rightarrow (\alpha \Rightarrow \text{bool}) \Rightarrow \text{bool})\) where
rel-reflects-pred \(\alpha\) \(\beta\) \(\epsilon\) \(\eta\) \(\zeta\) \(\eta\)

abbreviation rel-reflects-binary-pred :: \(((\alpha \times \alpha) \set \Rightarrow (\alpha \Rightarrow \text{bool}) \Rightarrow \text{bool})\) where
rel-reflects-binary-pred \(\alpha\) \(\beta\) \(\epsilon\) \(\eta\) \(\zeta\) \(\eta\)

A relation respects a predicate if it preserves and reflects it.

abbreviation rel-respects-pred :: \(((\alpha \times \alpha) \set \Rightarrow (\alpha \Rightarrow \text{bool}) \Rightarrow \text{bool})\) where
rel-respects-pred \(\alpha\) \(\beta\) \(\epsilon\) \(\eta\) \(\zeta\) \(\eta\)

abbreviation rel-respects-binary-pred :: \(((\alpha \times \alpha) \set \Rightarrow (\alpha \Rightarrow \text{bool}) \Rightarrow \text{bool})\) where
rel-respects-binary-pred \(\alpha\) \(\beta\) \(\epsilon\) \(\eta\) \(\zeta\) \(\eta\)

For symmetric relations preservation, reflection, and respection of predicates means the same.

lemma symm-relation-impl-preservation-equals-reflection:
fixes \(\alpha\) \(\beta\) \(\epsilon\) \(\eta\) \(\zeta\) \(\eta\)
and \(\gamma\) \(\delta\) \(\epsilon\) \(\zeta\) \(\eta\) \(\zeta\)
assumes \(\eta\) \(\zeta\) \(\eta\) \(\zeta\) \(\eta\) \(\zeta\)
shows \(\text{rel-preserves-pred} \text{ rel-preserves-binary-pred} \text{ rel-reflects-pred} \text{ rel-reflects-binary-pred}\)
using \(\eta\) \(\zeta\) \(\eta\) \(\zeta\) \(\eta\) \(\zeta\)
by (auto simp add: sym-def)

lemma symm-relation-impl-preservation-equals-reflection-of-binary-predicates:
fixes \(\alpha\) \(\beta\) \(\epsilon\) \(\eta\) \(\zeta\) \(\eta\)
and \(\gamma\) \(\delta\) \(\epsilon\) \(\zeta\) \(\eta\) \(\zeta\)
assumes \(\eta\) \(\zeta\) \(\eta\) \(\zeta\) \(\eta\) \(\zeta\)
shows \(\text{rel-preserves-binary-pred} \text{ rel-reflects-binary-pred}\)

and rel-preserves-binary-pred Rel Pred = rel-respects-binary-pred Rel Pred
and rel-reflects-binary-pred Rel Pred = rel-respects-binary-pred Rel Pred

using symm
unfolding sym-def
by blast+

If a relation preserves a predicate then so does its reflexive or/and transitive closure.

lemma preservation-and-closures:
fixes Rel :: ('a × 'a) set
and Pred :: 'a ⇒ bool
assumes preservation: rel-preserves-pred Rel Pred
shows rel-preserves-pred (Rel=) Pred
and rel-preserves-pred (Rel+) Pred
and rel-preserves-pred (Rel∗) Pred

proof –
from preservation show A: rel-preserves-pred (Rel=) Pred
by (auto simp add: refl)
have B: ∨Rel. rel-preserves-pred Rel Pred ⇒ rel-preserves-pred (Rel+) Pred
proof clarify
fix Rel a b
assume (a, b) ∈ Rel+ and rel-preserves-pred Rel Pred and Pred a
thus Pred b
by (induct, blast+)
qed
with preservation show rel-preserves-pred (Rel+) Pred
by blast
from preservation A B[where Rel=Rel=] show rel-preserves-pred (Rel∗) Pred
using trancl-reflcl[of Rel]
by blast
qed

lemma preservation-of-binary-predicates-and-closures:
fixes Rel :: ('a × 'a × 'a) set
and Pred :: 'a ⇒ 'b ⇒ bool
assumes preservation: rel-preserves-binary-pred Rel Pred
shows rel-preserves-binary-pred (Rel=) Pred
and rel-preserves-binary-pred (Rel+) Pred
and rel-preserves-binary-pred (Rel∗) Pred

proof –
from preservation show A: rel-preserves-binary-pred (Rel=) Pred
by (auto simp add: refl)
have B: ∨Rel. rel-preserves-binary-pred Rel Pred ⇒ rel-preserves-binary-pred (Rel+) Pred
proof clarify
fix Rel a b x
assume (a, b) ∈ Rel+ and rel-preserves-binary-pred Rel Pred and Pred a x
thus Pred b x
by (induct, blast+)
qed
with preservation show rel-preserves-binary-pred (Rel+) Pred
by blast
from preservation A B[where Rel=Rel=]
show rel-preserves-binary-pred (Rel∗) Pred
using trancl-reflcl[of Rel]
by fast
qed

If a relation reflects a predicate then so does its reflexive or/and transitive closure.

lemma reflection-and-closures:
fixes Rel :: ('a × 'a) set

...
and Pred :: 'a ⇒ bool
assumes reflection: rel-reflects-pred Rel Pred
shows rel-reflects-pred (Rel=) Pred
  and rel-reflects-pred (Rel+) Pred
  and rel-reflects-pred (Rel*) Pred

proof –
from reflection show A: rel-reflects-pred (Rel=) Pred
  by (auto simp add: refl)
have B: ∃ Rel. rel-reflects-pred Rel Pred ⇒ rel-reflects-pred (Rel+) Pred
proof clarify
  fix Rel a b
  assume (a, b) ∈ Rel+ and rel-reflects-pred Rel Pred and Pred b
  thus Pred a
  by (induct, blast+)
qed
with reflection show rel-reflects-pred (Rel+) Pred
  by blast
from reflection A B[where Rel=Rel=] show rel-reflects-pred (Rel*) Pred
  using trancl-reflcl[of Rel]
  by fast
qed

lemma reflection-of-binary-predicates-and-closures:
fixes Rel :: ('a × 'a) set
  and Pred :: 'a ⇒ 'b ⇒ bool
assumes reflection: rel-reflects-binary-pred Rel Pred
shows rel-reflects-binary-pred (Rel=) Pred
  and rel-reflects-binary-pred (Rel+) Pred
  and rel-reflects-binary-pred (Rel*) Pred
proof –
from reflection show A: rel-reflects-binary-pred (Rel=) Pred
  by (auto simp add: refl)
have B: ∃ Rel. rel-reflects-binary-pred Rel Pred ⇒ rel-reflects-binary-pred (Rel+) Pred
proof clarify
  fix Rel a b x
  assume (a, b) ∈ Rel+ and rel-reflects-binary-pred Rel Pred and Pred b x
  thus Pred a x
  by (induct, blast+)
qed
with reflection show rel-reflects-binary-pred (Rel+) Pred
  by blast
from reflection A B[where Rel=Rel=]
show rel-reflects-binary-pred (Rel*) Pred
  using trancl-reflcl[of Rel]
  by fast
qed

If a relation respects a predicate then so does its reflexive, symmetric, or/and transitive closure.

lemma respection-and-closures:
fixes Rel :: ('a × 'a) set
  and Pred :: 'a ⇒ bool
assumes respection: rel-respects-pred Rel Pred
shows rel-respects-pred (Rel=) Pred
  and rel-respects-pred (symcl Rel) Pred
  and rel-respects-pred (Rel+) Pred
  and rel-respects-pred (symcl (Rel=)) Pred
  and rel-respects-pred (Rel*) Pred
  and rel-respects-pred ((symcl (Rel=))+) Pred
proof –
from respection show A: rel-respects-pred (Rel=) Pred
  using preservation-and-closures(1)[where Rel=Rel and Pred=Pred]
reflection-and-closures(1)\[\textbf{where } \text{Rel} = \text{Rel} \text{ and } \text{Pred} = \text{Pred}\] by blast

have \(B : \forall \text{Rel. rel-respects-pred Rel Pred} \implies \text{rel-respects-pred (symcl Rel) Pred}\)

proof
fix \text{Rel}
assume \(B1 : \text{rel-respects-pred Rel Pred}\)
show \(\text{rel-preserves-pred (symcl Rel) Pred}\)
proof clarify
fix \(a \ b\)
assume \((a, b) \in \text{symcl Rel}\)

hence \((a, b) \in \text{Rel} \lor (b, a) \in \text{Rel}\)
by (simp add: symcl-def)

moreover assume \(\text{Pred} \ a\)
ultimately show \(\text{Pred} \ b\)
using \(B1\)
by blast
qed

next
fix \text{Rel} :: ('a \times 'a) set
and \text{Pred} :: 'a \Rightarrow bool
assume \(B2 : \text{rel-respects-pred Rel Pred}\)
show \(\text{rel-reflects-pred (symcl Rel) Pred}\)
proof clarify
fix \(a \ b\)
assume \((a, b) \in \text{symcl Rel}\)

hence \((a, b) \in \text{Rel} \lor (b, a) \in \text{Rel}\)
by (simp add: symcl-def)

moreover assume \(\text{Pred} \ b\)
ultimately show \(\text{Pred} \ a\)
using \(B2\)
by blast
qed

from respection \(B[\textbf{where } \text{Rel} = \text{Rel}]\) show \(\text{rel-respects-pred (symcl Rel) Pred}\)
by blast

have \(\forall \text{Rel. rel-respects-pred Rel Pred} \implies \text{rel-respects-pred (Rel\(\Rightarrow\)) Pred}\)
proof –
fix \text{Rel}
assume \(\text{rel-respects-pred Rel Pred}\)
thus \(\text{rel-respects-pred (Rel\(\Rightarrow\)) Pred}\)
using preservation-and-closures(2)[\textbf{where } \text{Rel} = \text{Rel} \text{ and } \text{Pred} = \text{Pred}] reflection-and-closures(2)[\textbf{where } \text{Rel} = \text{Rel} \text{ and } \text{Pred} = \text{Pred}] by blast

qed

from respection \(C[\textbf{where } \text{Rel} = \text{Rel}]\) show \(\text{rel-respects-pred (Rel\(\Rightarrow\)) Pred}\)
by blast
from \(A \ B[\textbf{where } \text{Rel} = \text{Rel}]\) show \(\text{rel-respects-pred (symcl (Rel\(\Rightarrow\))) Pred}\)
by blast
from \(A \ C[\textbf{where } \text{Rel} = \text{Rel}]\) show \(\text{rel-respects-pred (Rel\(\Rightarrow\)) Pred}\)
using trancl-reflcl[of \text{Rel}]
by fast
from \(A \ B[\textbf{where } \text{Rel} = \text{Rel}\Rightarrow]\) \(C[\textbf{where } \text{Rel} = \text{symcl (Rel\(\Rightarrow\))}]\)
show \(\text{rel-respects-pred ((symcl (Rel\(\Rightarrow\)))\(\Rightarrow\)) Pred}\)
by blast

qed

lemma respection-of-binary-predicates-and-closures:
fixes \text{Rel} :: ('a \times 'a) set
and \text{Pred} :: 'a \Rightarrow 'b \Rightarrow bool
assumes respection: \(\text{rel-respects-binary-pred Rel Pred}\)
shows \(\text{rel-respects-binary-pred (Rel\(\Rightarrow\)) Pred}\)
and rel-respects-binary-pred (symcl Rel) Pred
and rel-respects-binary-pred (Rel*) Pred
and rel-respects-binary-pred (symcl (Rel^=)) Pred
and rel-respects-binary-pred (Rel^+) Pred
and rel-respects-binary-pred ((symcl (Rel^=))^) Pred

proof –
from respection show A: rel-respects-binary-pred (Rel^=) Pred
  using preservation-of-binary-predicates-and-closures(1)[where Rel=Rel and Pred=Pred]
  reflection-of-binary-predicates-and-closures(1)[where Rel=Rel and Pred=Pred]
by blast
have B: \(\forall Rel. \) rel-respects-binary-pred Rel Pred \(\implies\) rel-respects-binary-pred (symcl Rel) Pred
proof
  fix Rel
  assume B1: rel-respects-binary-pred Rel Pred
  show rel-preserves-binary-pred (symcl Rel) Pred
  proof
clarify
  fix a b x
  assume \((a, b) \in \text{symcl Rel}\)
hence \((a, b) \in \text{Rel} \lor (b, a) \in \text{Rel}\)
  by (simp add: symcl-def)
moreover assume Pred a x
ultimately show Pred b x
  using B1
  by blast
qed
next
  fix Rel
  assume B2: rel-respects-binary-pred Rel Pred
  show rel-reflects-binary-pred (symcl Rel) Pred
  proof
clarify
  fix a b x
  assume \((a, b) \in \text{symcl Rel}\)
hence \((a, b) \in \text{Rel} \lor (b, a) \in \text{Rel}\)
  by (simp add: symcl-def)
moreover assume Pred b x
ultimately show Pred a x
  using B2
  by blast
qed
from respection B[where Rel=Rel] show rel-respects-binary-pred (symcl Rel) Pred
by blast
have C: \(\forall Rel. \) rel-respects-binary-pred Rel Pred \(\implies\) rel-respects-binary-pred (Rel^+) Pred
proof –
  fix Rel
  assume rel-respects-binary-pred Rel Pred
  thus rel-respects-binary-pred (Rel^+) Pred
    using preservation-of-binary-predicates-and-closures(2)[where Rel=Rel and Pred=Pred]
    reflection-of-binary-predicates-and-closures(2)[where Rel=Rel and Pred=Pred]
  by blast
qed
from respection C[where Rel=Rel] show rel-respects-binary-pred (Rel^+) Pred
  using trancl-reflcl[\text{of Rel}]
  by fast
from A B[where Rel=Rel^=] show rel-respects-binary-pred (symcl (Rel^=)) Pred
  by blast
from A C[where Rel=Rel^=] show rel-respects-binary-pred (Rel^+) Pred
  using trancl-reflcl[\text{of Rel}]
  by fast
from A B[where Rel=Rel^=] C[where Rel=symcl (Rel^=)]
2 Process Calculi

A process calculus is given by a set of process terms (syntax) and a relation on terms (semantics). We consider reduction as well as labelled variants of the semantics.

2.1 Reduction Semantics

A set of process terms and a relation on pairs of terms (called reduction semantics) define a process calculus.

```
record 'proc processCalculus =
  Reductions :: 'proc ⇒ 'proc ⇒ bool

A pair of the reduction relation is called a (reduction) step.

abbreviation step :: 'proc ⇒ 'proc processCalculus ⇒ 'proc ⇒ bool
  (− −→ − [70, 70, 70] 80)
  where
  P −→ Cal Q ≡ Reductions Cal P Q

We use * to indicate the reflexive and transitive closure of the reduction relation.

primrec nSteps :: 'proc ⇒ 'proc processCalculus ⇒ nat ⇒ 'proc ⇒ bool
  (− −→ − [70, 70, 70] 80)
  where
  P −→ Cal^0 Q = (P = Q) | P −→ Cal^Suc n Q = (∃ P'. P −→ Cal^n P' ∧ P' −→ Cal Q)

definition steps :: 'proc ⇒ 'proc processCalculus ⇒ 'proc ⇒ bool
  (− −→ * − [70, 70, 70] 80)
  where
  P −→ Cal* Q ≡ ∃ n. P −→ Cal^n Q
```

A process is divergent, if it can perform an infinite sequence of steps.

```
definition divergent :: 'proc ⇒ 'proc processCalculus ⇒ bool
  (− −→ ω [70, 70, 70] 80)
  where
  P −→ (Cal)ω ≡ ∀ P'. P −→ Cal* P' −→ (∃ P''. P' −→ Cal P'')
```

Each term can perform an (empty) sequence of steps to itself.

```
lemma steps-refl:
  fixes Cal :: 'proc processCalculus
  and P :: 'proc
  shows P −→ Cal^0 P
  proof −
    have P −→ Cal^0 P
      by simp
    hence ∃ n. P −→ Cal^n P
```
A single step is a sequence of steps of length one.

**lemma** step-to-steps:
```plaintext
fixes Cal :: 'proc processCalculus
    and P P' :: 'proc
assumes step: P →→ Cal P'
shows P →→ Cal* P'
```
```plaintext
proof –
from step have P →→ Cal P'
    by simp
thus thesis
    unfolding steps-def
    by blast
qed
```

If there is a sequence of steps from P to Q and from Q to R, then there is also a sequence of steps from P to R.

**lemma** nSteps-add:
```plaintext
fixes Cal :: 'proc processCalculus
    and n1 n2 :: nat
shows ∀ P Q R. P →→ Cal^n1 Q ∧ Q →→ Cal^n2 R →→ P →→ Cal^(n1 + n2) R
```
```plaintext
proof (induct n2, simp)
case (Suc n)
assume IH: ∀ P Q R. P →→ Cal^n1 Q ∧ Q →→ Cal^n R →→ P →→ Cal^(n1 + n) R
show ?case
proof clarify
fix P Q R
assume Q →→ Cal^{Suc n} R
from this obtain Q' where A1: Q →→ Cal^n Q' and A2: Q' →→ Cal R
    by auto
assume P →→ Cal^n1 Q
with A1 IH have P →→ Cal^(n1 + n) Q'
    by blast
with A2 show P →→ Cal^{n1 + Suc n} R
    by auto
qed
```
```plaintext
qed
```

**lemma** steps-add:
```plaintext
fixes Cal :: 'proc processCalculus
    and P Q R :: 'proc
assumes A1: P →→ Cal^n1 Q and A2: Q →→ Cal^n R
shows P →→ Cal^n R
```
```plaintext
proof –
from A1 obtain n1 where P →→ Cal^n1 Q
    by (auto simp add: steps-def)
moreover from A2 obtain n2 where Q →→ Cal^n2 R
    by (auto simp add: steps-def)
ultimately have P →→ Cal^(n1 + n2) R
    using nSteps-add[where Cal=Cal]
    by blast
thus P →→ Cal*R
    by (simp add: steps-def, blast)
qed
```
2.1.1 Observables or Barbs

We assume a predicate that tests terms for some kind of observables. At this point we do not limit or restrict the kind of observables used for a calculus nor the method to check them.

record ('proc, 'barbs) calculusWithBarbs =
  Calculus :: 'proc processCalcualus
  HasBarb :: 'proc ⇒ 'barbs ⇒ bool (-↓- [70, 70] 80)

abbreviation hasBarb
  :: 'proc ⇒ ('proc, 'barbs) calculusWithBarbs ⇒ 'barbs ⇒ bool
  (-↓<->- [70, 70] 80)
  where
  P↓<CWB>a ≡ HasBarb CWB P a

A term reaches a barb if it can evolve to a term that has this barb.

abbreviation reachesBarb
  :: ('proc × 'proc) set ⇒ ('proc, 'barbs) calculusWithBarbs ⇒ 'barbs ⇒ bool
  (-⇓<->- [70, 70], 70] 80)
  where
  P⇓<CWB>a ≡ ∃ P'. P ⇒→ (Calculus CWB)' P' ∧ P⇓<CWB>a

A relation R preserves barbs if whenever (P, Q) in R and P has a barb then also Q has this barb.

abbreviation rel-preserves-barb-set
  :: ('proc × 'proc) set ⇒ ('proc, 'barbs) calculusWithBarbs ⇒ 'barbs set ⇒ bool
  where
  rel-preserves-barb-set Rel CWB Barbs ≡ rel-preserves-binary-pred Rel (λP a. a ∈ Barbs ∧ P↓<CWB>a)

abbreviation rel-preserves-barbs
  :: ('proc × 'proc) set ⇒ ('proc, 'barbs) calculusWithBarbs ⇒ bool
  where
  rel-preserves-barbs Rel CWB ≡ rel-preserves-binary-pred Rel (HasBarb CWB)

lemma preservation-of-barbs-and-set-of-barbs:
  fixes Rel :: ('proc × 'proc) set
  and CWB :: ('proc, 'barbs) calculusWithBarbs
  shows rel-preserves-barbs Rel CWB = (∀ Barbs. rel-preserves-barb-set Rel CWB Barbs)
  by blast

A relation R reflects barbs if whenever (P, Q) in R and P has a barb then also Q has this barb.

abbreviation rel-reflects-barb-set
  :: ('proc × 'proc) set ⇒ ('proc, 'barbs) calculusWithBarbs ⇒ 'barbs set ⇒ bool
  where
  rel-reflects-barb-set Rel CWB Barbs ≡ rel-reflects-binary-pred Rel (λP a. a ∈ Barbs ∧ P⇓<CWB>a)

abbreviation rel-reflects-barbs
  :: ('proc × 'proc) set ⇒ ('proc, 'barbs) calculusWithBarbs ⇒ bool
  where
  rel-reflects-barbs Rel CWB ≡ rel-reflects-binary-pred Rel (HasBarb CWB)

lemma reflection-of-barbs-and-set-of-barbs:
  fixes Rel :: ('proc × 'proc) set
  and CWB :: ('proc, 'barbs) calculusWithBarbs
  shows rel-reflects-barbs Rel CWB = (∀ Barbs. rel-reflects-barb-set Rel CWB Barbs)
  by blast

A relation respects barbs if it preserves and reflects barbs.

abbreviation rel-respects-barb-set
(: ('proc × 'proc) set ⇒ ('proc, 'barbs) calculusWithBarbs ⇒ 'barbs set ⇒ bool

where
rel-respects-barb-set Rel CWB Barbs ≡
rel-preserves-barb-set Rel CWB Barbs ∧ rel-reflects-barb-set Rel CWB Barbs

abbreviation rel-respects-barbs
:: ('proc × 'proc) set ⇒ ('proc, 'barbs) calculusWithBarbs ⇒ bool

where
rel-respects-barbs Rel CWB ≡ rel-preserves-barb-set Rel CWB Barbs

lemma respection-of-barbs-and-set-of-barbs:
fixes Rel :: ('proc × 'proc) set
and CWB :: ('proc, 'barbs) calculusWithBarbs
shows rel-respects-barbs Rel CWB = (∀ Barbs. rel-respects-barb-set Rel CWB Barbs)
by blast

If a relation preserves barbs then so does its reflexive or/and transitive closure.

lemma preservation-of-barbs-and-closures:
fixes Rel :: ('proc × 'proc) set
and CWB :: ('proc, 'barbs) calculusWithBarbs
assumes preservation: rel-preserves-barbs Rel CWB
shows rel-preserves-barbs (Rel") CWB
and rel-preserves-barbs (Rel") CWB
and rel-preserves-barbs (Rel") CWB
using preservation
preservation-of-binary-predicates-and-closures
[where Rel=Rel and Pred=HasBarb CWB]
by blast

If a relation reflects barbs then so does its reflexive or/and transitive closure.

lemma reflection-of-barbs-and-closures:
fixes Rel :: ('proc × 'proc) set
and CWB :: ('proc, 'barbs) calculusWithBarbs
assumes reflection: rel-reflects-barbs Rel CWB
shows rel-reflects-barbs (Rel") CWB
and rel-reflects-barbs (Rel") CWB
and rel-reflects-barbs (Rel") CWB
using reflection
reflection-of-binary-predicates-and-closures
[where Rel=Rel and Pred=HasBarb CWB]
by blast

If a relation respects barbs then so does its reflexive, symmetric, or/and transitive closure.

lemma respection-of-barbs-and-closures:
fixes Rel :: ('proc × 'proc) set
and CWB :: ('proc, 'barbs) calculusWithBarbs
assumes respection: rel-respects-barbs Rel CWB
shows rel-respects-barbs (Rel") CWB
and rel-respects-barbs (symcl Rel") CWB
and rel-respects-barbs (symcl (Rel")") CWB
and rel-respects-barbs (Rel") CWB
and rel-respects-barbs (symcl (Rel")") CWB
proof
from respection show rel-respects-barbs (Rel") CWB
using respection-of-binary-predicates-and-closures(1)
[where Rel=Rel and Pred=HasBarb CWB]
by blast
next
from respection show rel-respects-barbs (symcl Rel") CWB
using respection-of-binary-predicates-and-closures(2)
[where Rel=Rel and Pred=HasBarb CWB]
by blast
next
from respection show rel-respects-barbs (Rel⁺) CWB
using respection-of-binary-predicates-and-closures(3)[where Rel=Rel and Pred=HasBarb CWB]
by blast
next
from respection show rel-respects-barbs (symcl (Rel⁻)) CWB
using respection-of-binary-predicates-and-closures(4)[where Rel=Rel and Pred=HasBarb CWB]
by blast
next
from respection show rel-respects-barbs (Rel*) CWB
using respection-of-binary-predicates-and-closures(5)[where Rel=Rel and Pred=HasBarb CWB]
by blast
next
from respection show rel-respects-barbs ((symcl (Rel⁺))⁺) CWB
using respection-of-binary-predicates-and-closures(6)[where Rel=Rel and Pred=HasBarb CWB]
by blast
qed

A relation R weakly preserves barbs if it preserves reachability of barbs, i.e., if (P, Q) in R and P reaches a barb then also Q has to reach this barb.

abbreviation rel-weakly-preserves-barb-set :: (′proc × ′proc) set ⇒ (′proc, ′barbs) calculusWithBarbs ⇒ bool
where
rel-weakly-preserves-barb-set Rel CWB Barbs ≡ rel-preserves-binary-pred Rel (λP a. a ∈ Barbs ∧ P⇓<CWB>a)

abbreviation rel-weakly-preserves-barbs :: (′proc × ′proc) set ⇒ (′proc, ′barbs) calculusWithBarbs ⇒ bool
where
rel-weakly-preserves-barbs Rel CWB ≡ rel-weakly-preserves-barb-set Rel CWB Barbs

lemma weak-preservation-of-barbs-and-set-of-barbs:
fixes Rel :: (′proc × ′proc) set
and CWB :: (′proc, ′barbs) calculusWithBarbs
shows rel-weakly-preserves-barbs Rel CWB
= (∀ Barbs. rel-weakly-preserves-barb-set Rel CWB Barbs)
by blast

A relation R weakly reflects barbs if it reflects reachability of barbs, i.e., if (P, Q) in R and Q reaches a barb then also P has to reach this barb.

abbreviation rel-weakly-reflects-barb-set :: (′proc × ′proc) set ⇒ (′proc, ′barbs) calculusWithBarbs ⇒ bool
where
rel-weakly-reflects-barb-set Rel CWB Barbs ≡ rel-reflects-binary-pred Rel (λP a. a ∈ Barbs ∧ P⇓<CWB>a)

abbreviation rel-weakly-reflects-barbs :: (′proc × ′proc) set ⇒ (′proc, ′barbs) calculusWithBarbs ⇒ bool
where
rel-weakly-reflects-barbs Rel CWB ≡ rel-weakly-reflects-barb-set Rel CWB Barbs

lemma weak-reflection-of-barbs-and-set-of-barbs:
fixes Rel :: (′proc × ′proc) set
and CWB :: (′proc, ′barbs) calculusWithBarbs
shows rel-weakly-reflects-barbs Rel CWB = (∀ Barbs. rel-weakly-reflects-barb-set Rel CWB Barbs)
by blast

A relation weakly respects barbs if it weakly preserves and weakly reflects barbs.

abbreviation rel-weakly-respects-barb-set :: (′proc × ′proc) set ⇒ (′proc, ′barbs) calculusWithBarbs ⇒ bool
where
rel-weakly-respects-barb-set Rel CWB Barbs ≡ rel-weakly-preserves-barb-set Rel CWB Barbs ∧ rel-weakly-reflects-barb-set Rel CWB Barbs
\[
\text{rel-weakly-respects-barb-set } \text{Rel CWB Barbs} \equiv \\
\text{rel-weakly-preserves-barb-set } \text{Rel CWB Barbs} \land \text{rel-weakly-reflects-barb-set } \text{Rel CWB Barbs}
\]

**abbreviation** \text{rel-weakly-respects-barbs} :: \((\text{proc} \times \text{proc}) \text{ set} \Rightarrow (\text{proc}, \text{barbs}) \text{ calculusWithBarbs} \Rightarrow \text{bool}

**where**

\text{rel-weakly-respects-barbs } \text{Rel CWB} \equiv \\
\text{rel-weakly-preserves-barbs } \text{Rel CWB} \land \text{rel-weakly-reflects-barbs } \text{Rel CWB}

**lemma** weak-respection-of-barbs-and-set-of-barbs:

- **fixes** \text{Rel} :: \((\text{proc} \times \text{proc}) \text{ set}
- **and** \text{CWB} :: \((\text{proc}, \text{barbs}) \text{ calculusWithBarbs}
- **shows** \text{rel-weakly-respects-barbs } \text{Rel CWB} = (\forall \text{Barbs. rel-weakly-respects-barb-set Rel CWB Barbs)
- **by** blast

If a relation weakly preserves barbs then so does its reflexive or/and transitive closure.

**lemma** weak-preservation-of-barbs-and-closures:

- **fixes** \text{Rel} :: \((\text{proc} \times \text{proc}) \text{ set}
- **and** \text{CWB} :: \((\text{proc}, \text{barbs}) \text{ calculusWithBarbs}
- **assumes** preservation: \text{rel-weakly-preserves-barbs } \text{Rel CWB}
- **shows** \text{rel-weakly-preserves-barbs } (\text{Rel}^{=}) \text{ CWB}
- **and** \text{rel-weakly-preserves-barbs } (\text{Rel}^{+}) \text{ CWB}
- **and** \text{rel-weakly-preserves-barbs } (\text{Rel}^{*}) \text{ CWB}
- **using** preservation preservation-of-binary-predicates-and-closures[where \text{Rel}=\text{Rel}
- **and** \text{Pred}=\lambda P a. P|<\text{CWB}>a]
- **by** blast+

If a relation weakly reflects barbs then so does its reflexive or/and transitive closure.

**lemma** weak-reflection-of-barbs-and-closures:

- **fixes** \text{Rel} :: \((\text{proc} \times \text{proc}) \text{ set}
- **and** \text{CWB} :: \((\text{proc}, \text{barbs}) \text{ calculusWithBarbs}
- **assumes** reflection: \text{rel-weakly-reflects-barbs } \text{Rel CWB}
- **shows** \text{rel-weakly-reflects-barbs } (\text{Rel}^{=}) \text{ CWB}
- **and** \text{rel-weakly-reflects-barbs } (\text{Rel}^{+}) \text{ CWB}
- **and** \text{rel-weakly-reflects-barbs } (\text{Rel}^{*}) \text{ CWB}
- **using** reflection reflection-of-binary-predicates-and-closures[where \text{Rel}=\text{Rel}
- **and** \text{Pred}=\lambda P a. P|<\text{CWB}>a]
- **by** blast+

If a relation weakly respects barbs then so does its reflexive, symmetric, or/and transitive closure.

**lemma** weak-respection-of-barbs-and-closures:

- **fixes** \text{Rel} :: \((\text{proc} \times \text{proc}) \text{ set}
- **and** \text{CWB} :: \((\text{proc}, \text{barbs}) \text{ calculusWithBarbs}
- **assumes** respection: \text{rel-weakly-respects-barbs } \text{Rel CWB}
- **shows** \text{rel-weakly-respects-barbs } (\text{Rel}^{=}) \text{ CWB}
- **and** \text{rel-weakly-respects-barbs } (\text{symcl Rel}) \text{ CWB}
- **and** \text{rel-weakly-respects-barbs } (\text{Rel}^{+}) \text{ CWB}
- **and** \text{rel-weakly-respects-barbs } (\text{symcl (Rel}^{=})) \text{ CWB}
- **and** \text{rel-weakly-respects-barbs } (\text{symcl (Rel}^{+})) \text{ CWB}
- **using** respection-of-binary-predicates-and-closures[where \text{Rel}=\text{Rel}
- **and** \text{Pred}=\lambda P a. P|<\text{CWB}>a]
- **by** blast+

**proof**

- **from** respection **show** \text{rel-weakly-respects-barbs } (\text{Rel}^{=}) \text{ CWB}
- **using** respection-of-binary-predicates-and-closures(1)[where \text{Rel}=\text{Rel}
- **and** \text{Pred}=\lambda P a. P|<\text{CWB}>a]
- **by** blast

**next**

- **from** respection **show** \text{rel-weakly-respects-barbs } (\text{symcl Rel}) CWB
- **using** respection-of-binary-predicates-and-closures(2)[where \text{Rel}=\text{Rel}
- **and** \text{Pred}=\lambda P a. P|<\text{CWB}>a]
- **by** blast
3 Simulation Relations

Simulation relations are a special kind of property on relations on processes. They usually require that steps are (strongly or weakly) preserved and/or reflected modulo the relation. We consider different kinds of simulation relations.

3.1 Simulation

A weak reduction simulation is relation \( R \) such that if \((P, Q) \in R\) and \( P \) evolves to some \( P' \) then there exists some \( Q' \) such that \( Q \) evolves to \( Q' \) and \((P', Q') \in R\).

**abbreviation** weak-reduction-simulation

\[ \overset{3}{\text{proc} \times \text{proc}} \rightarrow \text{proc processCalculus} \Rightarrow \text{bool} \]

**where**

\[
\text{weak-reduction-simulation} \ (R) \equiv \\
\forall P, Q, P'. (P, Q) \in R \land P \rightarrow Cal \ P' \rightarrow (\exists Q'. Q \rightarrow Cal \ Q' \land (P', Q') \in R)
\]

A weak barbed simulation is weak reduction simulation that weakly preserves barbs.

**abbreviation** weak-barbed-simulation

\[ \overset{3}{\text{proc} \times \text{proc}} \rightarrow (\text{proc}, \text{barbs}) \text{calculusWithBarbs} \Rightarrow \text{bool} \]

**where**

\[
\text{weak-barbed-simulation} \ (R) \equiv \\
\text{weak-reduction-simulation} \ (C) \land \text{rel-weakly-preserves-barbs} \ (R)
\]

The reflexive and/or transitive closure of a weak simulation is a weak simulation.

**lemma** weak-reduction-simulation-and-closures:

**fixes**

\[ (\text{proc} \times \text{proc}) \rightarrow (\text{proc}) \text{processCalculus} \]

**and**

\[ \text{Cal} :: (\text{proc}) \text{calculusWithBarbs} \]

**assumes**

\[ \text{simulation: weak-reduction-simulation} \ (R) \]

**shows**

\[ \text{weak-reduction-simulation} \ (R) \]

\[ \text{and weak-reduction-simulation} \ (R) \]

\[ \text{and weak-reduction-simulation} \ (R) \]
proof –
from simulation show A: weak-reduction-simulation (Rel⁺) Cal
  by (auto simp add: refl, blast)
have B: \( \forall Q \in \text{Rel} \). weak-reduction-simulation Rel Cal \implies weak-reduction-simulation (Rel⁺) Cal
proof clarify
  fix Q P P'
  assume B1: weak-reduction-simulation Rel Cal
  assume \( (P, Q) \in \text{Rel} \) and \( P \mapsto Cal^* P' \)
  thus \( \exists Q'. Q \mapsto Cal^* Q' \land (P', Q') \in \text{Rel} \)
proof (induct arbitrary: \( P' \))
  fix Q P'
  assume \( (P, Q) \in \text{Rel} \) and \( P \mapsto Cal^* P' \)
  with B1 obtain \( Q' \) where \( Q \mapsto Cal^* Q' \land (P', Q') \in \text{Rel} \)
    by blast
  thus \( \exists Q'. Q \mapsto Cal^* Q' \land (P', Q') \in \text{Rel} \)
    by auto
next
  case (step Q R P')
  assume \( \forall P'. P \mapsto Cal^* P' \implies (\exists Q'. Q \mapsto Cal^* Q' \land (P', Q') \in \text{Rel} \) and \( P \mapsto Cal^* P' \)
  from this obtain \( Q' \) where B2: \( Q \mapsto Cal^* Q' \land B3: (P', Q') \in \text{Rel} \)
    by blast
  assume \( (Q, R) \in \text{Rel} \)
  with B1 B2 obtain \( R' \) where B4: \( R \mapsto Cal^* R' \land B5: (Q', R') \in \text{Rel} \)
    by blast
  from B3 B5 have \( (P', R') \in \text{Rel} \)
    by simp
  from B4 this show \( \exists R'. R \mapsto Cal^* R' \land (P', R') \in \text{Rel} \)
    by blast
qed
qed
with simulation show weak-reduction-simulation (Rel⁺) Cal
  by blast
from simulation A B[where Rel=Rel⁺]
show weak-reduction-simulation (Rel⁺) Cal
  using trancl-refl[\text{of Rel}]
  by fast
qed

lemma weak-barbed-simulation-and-closures:
fixes Rel :: ('proc \
\times \ 'proc) set
  and CWB :: ('proc, 'barbs) calculusWithBars
assumes simulation: weak-barbed-simulation Rel CWB
shows weak-barbed-simulation (Rel⁺) CWB
  and weak-barbed-simulation (Rel⁺) CWB
  and weak-barbed-simulation (Rel⁺) CWB
proof –
from simulation show weak-barbed-simulation (Rel⁺) CWB
  using weak-reduction-simulation-and-closures(1)[where Rel=Rel and Cal=Calculus CWB]
  weak-preservation-of-barbs-and-closures(1)[where Rel=Rel and CWB=CWB]
    by blast
next
from simulation show weak-barbed-simulation (Rel⁺) CWB
  using weak-reduction-simulation-and-closures(2)[where Rel=Rel and Cal=Calculus CWB]
  weak-preservation-of-barbs-and-closures(2)[where Rel=Rel and CWB=CWB]
    by blast
next
from simulation show weak-barbed-simulation (Rel⁺) CWB
  using weak-reduction-simulation-and-closures(3)[where Rel=Rel and Cal=Calculus CWB]
  weak-preservation-of-barbs-and-closures(3)[where Rel=Rel and CWB=CWB]
    by blast
In the case of a simulation weak preservation of barbs can be replaced by the weaker condition that whenever \((P, Q)\) in the relation and \(P\) has a barb then \(Q\) have to be able to reach this barb.

**abbreviation** weak-barbed-preservation-cond
:: \( (\text{'proc } \times \text{'proc}) \text{ set } \Rightarrow \text{'proc, 'barbs} \text{ calculusWithBarbs } \Rightarrow \text{ bool} \)

where
weak-barbed-preservation-cond \(\text{Rel CWB} \equiv \forall P Q. a. \; (P, Q) \in \text{Rel } \land \; P \downarrow \text{CWB} > a \; \rightarrow \; Q \downarrow \text{CWB} > a\)

**lemma** weak-preservation-of-barbs:
fixes \(\text{Rel}:: (\text{'proc } \times \text{'proc}) \text{ set} \)
and \(\text{CWB}:: (\text{'proc, 'barbs}) \text{ calculusWithBarbs}\)
assumes preservation: rel-weakly-preserves-barbs \(\text{Rel CWB}\)
shows weak-barbed-preservation-cond \(\text{Rel CWB}\)

**proof**
clarify
fix \(P Q a\)
have \(P \mapsto \text{(Calculus CWB)}^* P\)
by \(\text{(simp add: steps-refl)}\)
moreover assume \(P \downarrow \text{CWB} > a\)
ultimately have \(P \downarrow \text{CWB} > a\)
by \text{blast}
moreover assume \((P, Q) \in \text{Rel}\)
ultimately show \(Q \downarrow \text{CWB} > a\)
using preservation
by \text{blast}

qed

**lemma** simulation-impl-equality-of-preservation-of-barbs-conditions:
fixes \(\text{Rel}:: (\text{'proc } \times \text{'proc}) \text{ set} \)
and \(\text{CWB}:: (\text{'proc, 'barbs}) \text{ calculusWithBarbs}\)
assumes simulation: weak-reduction-simulation \(\text{Rel (Calculus CWB)}\)
shows rel-weakly-preserves-barbs \(\text{Rel CWB} = \text{weak-barbed-preservation-cond Rel CWB}\)

**proof**
assume rel-weakly-preserves-barbs \(\text{Rel CWB}\)
thus weak-barbed-preservation-cond \(\text{Rel CWB}\)
using weak-preservation-of-barbs[where \(\text{Rel = Rel and CWB = CWB}\)]
by \text{blast}

next
assume condition: weak-barbed-preservation-cond \(\text{Rel CWB}\)
show rel-weakly-preserves-barbs \(\text{Rel CWB}\)

**proof**
clarify
fix \(P Q a\)
assume \((P, Q) \in \text{Rel and } P \mapsto \text{(Calculus CWB)}^* P'\)
with simulation obtain \(Q'\) where \(A1: Q \mapsto \text{(Calculus CWB)}^* Q'\) and \(A2: (P', Q') \in \text{Rel}\)
by \text{blast}
assume \(P' \downarrow \text{CWB} > a\)
with \(A2\) condition obtain \(Q''\) where \(A3: Q' \mapsto \text{(Calculus CWB)}^* Q''\) and \(A4: Q'' \downarrow \text{CWB} > a\)
by \text{blast}
from \(A1\) \(A2\) have \(Q \mapsto \text{(Calculus CWB)}^* Q''\)
by \text{(rule steps-add)}
with \(A4\) show \(Q \downarrow \text{CWB} > a\)
by \text{blast}

qed

A strong reduction simulation is relation \(R\) such that for each pair \((P, Q)\) in \(R\) and each step of \(P\) to some \(P'\) there exists some \(Q'\) such that there is a step of \(Q\) to \(Q'\) and \((P', Q')\) in \(R\).

**abbreviation** strong-reduction-simulation :: \( (\text{'proc } \times \text{'proc}) \text{ set } \Rightarrow \text{'proc processCalculus } \Rightarrow \text{ bool} \)

where
strong-reduction-simulation \(\text{Rel Cal} \equiv \)
∀ P Q P'. (P, Q) ∈ Rel ∧ P → Cal P' → (∃ Q'. Q → Cal Q' ∧ (P', Q') ∈ Rel)

A strong barbed simulation is strong reduction simulation that preserves barbs.

**abbreviation** strong-barbed-simulation
:: ('proc × 'proc) set ⇒ ('proc, 'barbs) calculusWithBarbs ⇒ bool

where
strong-barbed-simulation Rel CWB ≡
strong-reduction-simulation Rel (Calculus CWB) ∧ rel-preserves-barbs Rel CWB

A strong simulation is also a weak simulation.

**lemma** strong-impl-weak-reduction-simulation:

fixes Rel :: ('proc × 'proc) set
and Cal :: ('proc processCalculus)

assumes simulation: strong-reduction-simulation Rel Cal

shows weak-reduction-simulation Rel Cal

**proof** clarify

fix P Q P'
assume A1: (P, Q) ∈ Rel
assume P → Cal P'
from this obtain n where P → Cal^n P'
  by (auto simp add: steps-def)
thus ∃ Q'. Q → Cal Q' ∧ (P', Q') ∈ Rel

**proof** (induct n arbitrary: P')

**case** 0

assume P → Cal^0 P'

**hence** P = P'

Moreover have Q → Cal Q
  by (rule steps-refl)

ultimately show ∃ Q'. Q → Cal Q' ∧ (P', Q') ∈ Rel
  using A1
  by blast

**next**

**case** (Suc n P'')

assume P → Cal^Suc n P''
from this obtain P' where A2: P → Cal^n P' and A3: P' → Cal P''
  by auto
assume ∧ P', P → Cal^n P' ⇒ ∃ Q'. Q → Cal Q' ∧ (P', Q') ∈ Rel
with A2 obtain Q' where A4: Q → Cal Q' and A5: (P', Q') ∈ Rel
  by blast
from simulation A5 A3 obtain Q'' where A6: Q' → Cal Q'' and A7: (P'', Q'') ∈ Rel
  by blast
from A4 A6 have Q → Cal Q''
  using steps-add[where P=Q and Q=Q' and R=Q'']
    by (simp add: step-to-steps)
with A7 show ∃ Q'. Q → Cal Q' ∧ (P'', Q') ∈ Rel
  by blast

 qed

dq

dq

**lemma** strong-barbed-simulation-impl-weak-preservation-of-barbs:

fixes Rel :: ('proc × 'proc) set
and CWB :: ('proc, 'barbs) calculusWithBarbs

assumes simulation: strong-barbed-simulation Rel CWB

shows rel-weakly-preserves-barbs Rel CWB

**proof** clarify

fix P Q a P'
assume (P, Q) ∈ Rel and P → (Calculus CWB)* P'
with simulation obtain Q' where A1: Q → (Calculus CWB)* Q' and A2: (P', Q') ∈ Rel
  using strong-impl-weak-reduction-simulation[where Rel=Rel and Cal=Calculus CWB]
The reflexive and/or transitive closure of a strong simulation is a strong simulation.

**Lemma** strong-impl-weak-barbed-simulation:
- **Fixes** `Rel :: ('proc × 'proc) set`
- **Assumes** `simulation`: strong-barbed-simulation `Rel CWB`
- **Shows** `weak-barbed-simulation`: `Rel CWB`

**Proof**
- **From** simulation show `A`: strong-reduction-simulation `Rel Cal`
  - **By** (auto simp add: refl blast)
  - **Have** `B: ∀Rel. strong-reduction-simulation`: `Rel Cal`
    - **Imply** `strong-reduction-simulation`: `Rel Cal`

**Lemma** strong-reduction-simulation-and-closures:
- **Fixes** `Rel :: ('proc × 'proc) set`
- **Assumes** `simulation`: strong-reduction-simulation `Rel Cal`
- **Shows** `strong-reduction-simulation`: `Rel Cal`
- **And** `strong-reduction-simulation`: `Rel Cal`

**Proof**
- **Case** (step `Q R P'`)
  - **Assume** `∀P'. P' → Cal P' → (∃Q'. Q → Cal Q' ∧ (P', Q') ∈ Rel+)
  - **And** `P → Cal P'`
  - **From** this obtain `Q'` where `B2: Q → Cal Q'` and `B3: (P', Q') ∈ Rel`
    - **By** blast
  - **Thus** `∃Q'. Q → Cal Q' ∧ (P', Q') ∈ Rel+
    - **By** auto

**QED**
show strong-reduction-simulation (Rel\(^\ast\)) Cal
using trancl-refcl[of Rel]
by fast
qed

lemma strong-barbed-simulation-and-closures:
fixes Rel :: ('proc × 'proc) set
and CWB :: ('proc, 'barbs) calculusWithBarbs
assumes simulation: strong-barbed-simulation Rel CWB
shows strong-barbed-simulation (Rel\(^\ast\)) CWB
and strong-barbed-simulation (Rel\(^+\)) CWB
and strong-barbed-simulation (Rel\(=\)) CWB

proof
from simulation show strong-barbed-simulation (Rel\(=\)) CWB
using strong-reduction-simulation-and-closures(1)[where Rel=Rel and Cal=Calculus CWB]
preservation-of-barbs-and-closures(1)[where Rel=Rel and CWB=CWB]
by blast
next
from simulation show strong-barbed-simulation (Rel\(^+\)) CWB
using strong-reduction-simulation-and-closures(2)[where Rel=Rel and Cal=Calculus CWB]
preservation-of-barbs-and-closures(2)[where Rel=Rel and CWB=CWB]
by blast
next
from simulation show strong-barbed-simulation (Rel\(^\ast\)) CWB
using strong-reduction-simulation-and-closures(3)[where Rel=Rel and Cal=Calculus CWB]
preservation-of-barbs-and-closures(3)[where Rel=Rel and CWB=CWB]
by blast
qed

3.2 Contrasimulation

A weak reduction contrasimulation is relation \(R\) such that if \((P, Q)\) in \(R\) and \(P\) evolves to some \(P'\) then there exists some \(Q'\) such that \(Q\) evolves to \(Q'\) and \((Q', P')\) in \(R\).

abbreviation weak-reduction-contrasimulation
:: ('proc × 'proc) set ⇒ bool
where
weak-reduction-contrasimulation Rel Cal ≡
\(\forall P, Q, P'.\) \((P, Q)\) in Rel \(⇒\) \(P\) \(\rightarrow\) Cal \(⇒\) \(P'\) \(\rightarrow\) (\(\exists Q'.\) \(P\) \(\rightarrow\) Q \(⇒\) \(Q'\) \(⇒\) \((Q', P')\) in Rel)

A weak barbed contrasimulation is weak reduction contrasimulation that weakly preserves barbs.

abbreviation weak-barbed-contrasimulation
:: ('proc × 'proc) set ⇒ bool
where
weak-barbed-contrasimulation Rel CWB ≡
weak-reduction-contrasimulation (Calculus CWB) \(⇒\) rel-weakly-preserves-barbs Rel CWB

The reflexive and/or transitive closure of a weak contrasimulation is a weak contrasimulation.

lemma weak-reduction-contrasimulation-and-closures:
fixes Rel :: ('proc × 'proc) set
and Cal :: 'proc processCalculus
assumes contrasimulation: weak-reduction-contrasimulation Rel Cal
shows weak-reduction-contrasimulation (Rel\(^\ast\)) Cal
and weak-reduction-contrasimulation (Rel\(^+\)) Cal
and weak-reduction-contrasimulation (Rel\(\cdot\)) Cal

proof
from contrasimulation show A: weak-reduction-contrasimulation (Rel\(\cdot\)) Cal
by (auto simp add: refl, blast)
next
have B: \(\forall Rel.\) weak-reduction-contrasimulation Rel Cal
\(⇒\) weak-reduction-contrasimulation (Rel\(^+\)) Cal
proof clarify
fix Rel P Q P'
assume B1: weak-reduction-contrasimulation Rel Cal
assume (P, Q) ∈ Rel⁺ and P ⊢ Cal* P'
thus ∃ Q', Q ⊢ Cal* Q' ∧ (Q', P') ∈ Rel⁺
proof (induct arbitrary: P')
fix Q P'
assume (P, Q) ∈ Rel and P ⊢ Cal* P'
with B1 obtain Q' where Q ⊢ Cal* Q' and (Q', P') ∈ Rel
by blast
thus ∃ Q', Q ⊢ Cal* Q' ∧ (Q', P') ∈ Rel⁺
by auto
next
proof
thus ∃ (Q, R) ∈ Rel
with B1 B2 obtain R' where B4: R ⊢ Cal* R' and B5: (R', Q') ∈ Rel⁺
by blast
from this obtain Q' where B2: Q ⊢ Cal* Q' and B3: (Q', P') ∈ Rel⁺
by blast
assume (Q, R) ∈ Rel
with B1 B2 obtain R' where B4: R ⊢ Cal* R' and B5: (R', Q') ∈ Rel⁺
by blast
from B5 B3 have (R', P') ∈ Rel⁺
by simp
with B4 show ∃ R'. R ⊢ Cal* R' ∧ (R', P') ∈ Rel⁺
by blast
qed

with contrasimulation show weak-reduction-contrasimulation (Rel⁺) Cal
by blast
from contrasimulation A B[where Rel=Rel⁺]
show weak-reduction-contrasimulation (Rel⁺) Cal
using trancl-refcl[of Rel]
by fast
qed

lemma weak-barbed-contrasimulation-and-closures:
fixes Rel :: ('proc × 'proc) set
and CWB :: ('proc, 'barbs) calculusWithBarbs
assumes contrasimulation: weak-barbed-contrasimulation Rel CWB
shows weak-barbed-contrasimulation (Rel⁺) CWB
and weak-barbed-contrasimulation (Rel⁺) CWB
and weak-barbed-contrasimulation (Rel⁺) CWB
proof –
from contrasimulation show weak-barbed-contrasimulation (Rel⁺) CWB
using weak-reduction-contrasimulation-and-closures(1)[where Rel=Rel and Cal=Calculus CWB]
weak-preservation-of-barbs-and-closures(1)[where Rel=Rel and CWB=CWB]
by blast
next
from contrasimulation show weak-barbed-contrasimulation (Rel⁺) CWB
using weak-reduction-contrasimulation-and-closures(2)[where Rel=Rel and Cal=Calculus CWB]
weak-preservation-of-barbs-and-closures(2)[where Rel=Rel and CWB=CWB]
by blast
next
from contrasimulation show weak-barbed-contrasimulation (Rel⁺) CWB
using weak-reduction-contrasimulation-and-closures(3)[where Rel=Rel and Cal=Calculus CWB]
weak-preservation-of-barbs-and-closures(3)[where Rel=Rel and CWB=CWB]
by blast
qed
3.3 Coupled Simulation

A weak reduction coupled simulation is relation R such that if \((P, Q)\) in R and P evolves to some \(P'\) then there exists some \(Q'\) such that Q evolves to \(Q'\) and \((P', Q')\) in R and there exits some \(Q'\) such that Q evolves to \(Q'\) and \((Q', P')\) in R.

**abbreviation** weak-reduction-coupled-simulation
:: \((\text{'proc} \times \text{'proc})\) set ⇒ bool

where
weak-reduction-coupled-simulation Rel Cal ≡
\(\forall P Q P', (P, Q) \in \text{Rel} \land P \rightarrow Cal P' \rightarrow (\exists Q'. Q \rightarrow Cal Q' \land (P', Q') \in \text{Rel}) \land (\exists Q'. Q \rightarrow Cal Q' \land (Q', P') \in \text{Rel})\)

A weak barbed coupled simulation is weak reduction coupled simulation that weakly preserves barbs.

**abbreviation** weak-barbed-coupled-simulation
:: \((\text{'proc} \times \text{'proc})\) set ⇒ \((\text{'proc}, \text{'barbs})\) calculusWithBarbs ⇒ bool

where
weak-barbed-coupled-simulation Rel CWB ≡
weak-reduction-coupled-simulation Rel (Calculus CWB) \land \text{rel-weakly-preserves-barbs} Rel CWB

A weak coupled simulation combines the conditions on a weak simulation and a weak contrasimulation.

**lemma** weak-reduction-coupled-simulation-versus-simulation-and-contrasimulation:

fixes Rel :: \((\text{'proc} \times \text{'proc})\) set
and Cal :: \text{'proc processCalculus}
shows weak-reduction-coupled-simulation Rel Cal = (weak-reduction-simulation Rel Cal \land weak-reduction-contrasimulation Rel Cal)
by blast

**lemma** weak-barbed-coupled-simulation-versus-simulation-and-contrasimulation:

fixes Rel :: \((\text{'proc} \times \text{'proc})\) set
and CWB :: \((\text{'proc}, \text{'barbs})\) calculusWithBarbs
shows weak-barbed-coupled-simulation Rel CWB = (weak-barbed-simulation Rel CWB \land weak-barbed-contrasimulation Rel CWB)
by blast

The reflexive and/or transitive closure of a weak coupled simulation is a weak coupled simulation.

**lemma** weak-reduction-coupled-simulation-and-closures:

fixes Rel :: \((\text{'proc} \times \text{'proc})\) set
and Cal :: \text{'proc processCalculus}
assumes coupledSimulation: weak-reduction-coupled-simulation Rel Cal
shows weak-reduction-coupled-simulation \((\text{Rel}^+)\) Cal
and weak-reduction-coupled-simulation \((\text{Rel}^\ast)\) Cal
and weak-reduction-coupled-simulation \((\text{Rel}^\ast)\) Cal
using
weak-reduction-simulation-and-closures[where Rel=Rel and Cal=Cal]
weak-reduction-contrasimulation-and-closures[where Rel=Rel and Cal=Cal]
weak-reduction-coupled-simulation-versus-simulation-and-contrasimulation[where Rel=Rel and Cal=Cal]
coupledSimulation
by auto

**lemma** weak-barbed-coupled-simulation-and-closures:

fixes Rel :: \((\text{'proc} \times \text{'proc})\) set
and CWB :: \((\text{'proc}, \text{'barbs})\) calculusWithBarbs
assumes coupledSimulation: weak-barbed-coupled-simulation Rel CWB
shows weak-barbed-coupled-simulation \((\text{Rel}^+)\) CWB
and weak-barbed-coupled-simulation \((\text{Rel}^\ast)\) CWB
and weak-barbed-coupled-simulation \((\text{Rel}^\ast)\) CWB
proof –
from coupledSimulation show weak-barbed-coupled-simulation \((\text{Rel}^+)\) CWB
using weak-reduction-coupled-simulation-and-closures(1)[where Rel=Rel

24
3.4 Correspondence Simulation

A weak reduction correspondence simulation is relation \( R \) such that (1) if \((P, Q)\in R\) and \(P\) evolves to some \(P'\) then there exists some \(P''\) and \(Q''\) such that \(P\) evolves to \(P''\) and \(Q''\) evolves to \(Q''\) and \((P', Q')\in R\), and (2) if \((P, Q)\) in \(R\) and \(P\) evolves to some \(P'\) then there exists some \(Q'\) such that \(Q\) evolves to \(Q'\) and \((P', Q')\in R\).

**abbreviation** weak-reduction-correspondence-simulation

\[ \vdash (\text{'proc} \times \text{'proc})\text{ set} \Rightarrow (\text{'proc} \times \text{'proc})\text{ processCalculus} \Rightarrow \text{bool} \]

where

weak-reduction-correspondence-simulation \( R \) \( Cal \)

\[ (\forall P \ Q \ P'. (P, Q) \in R \land P \rightarrow C_\text{al*} P' \rightarrow (\exists Q'. Q \rightarrow C_\text{al*} Q' \land (P', Q') \in R)) \land (\forall P \ Q \ Q'. (P, Q) \in R \land Q \rightarrow C_\text{al*} Q' \rightarrow (\exists P'' Q''. P \rightarrow C_\text{al*} P'' \land Q' \rightarrow C_\text{al*} Q'' \land (P'', Q'') \in R)) \]

A weak barbed correspondence simulation is weak reduction correspondence simulation that weakly respects barbs.

**abbreviation** weak-barbed-correspondence-simulation

\[ \vdash (\text{'proc} \times \text{'proc})\text{ set} \Rightarrow (\text{'proc} \times \text{'barbs})\text{ calculusWithBarbs} \Rightarrow \text{bool} \]

where

weak-barbed-correspondence-simulation \( R \) \( CWB \)

\[ \land \text{rel-weakly-respects-barbs} \ R \ CWB \]

For each weak correspondence simulation \( R \) there exists a weak coupled simulation that contains all pairs of \( R \) in both directions.

**inductive-set** cSim-cs :: (\text{'proc} \times \text{'proc})\text{ set} \Rightarrow (\text{'proc} \times \text{'proc})\text{ processCalculus} \Rightarrow (\text{'proc} \times \text{'proc})\text{ set}

for Rel :: (\text{'proc} \times \text{'proc})\text{ set}

and Cal :: (\text{'proc} processCalculus)

where

left :: \([Q \rightarrow C_\text{al*} Q'; (P', Q') \in R]\) \Rightarrow (P', Q') \in cSim-cs Rel Cal

right :: \([P \rightarrow C_\text{al*} P'; (Q, P) \in R]\) \Rightarrow (P', Q) \in cSim-cs Rel Cal

trans :: \([(P, Q) \in C_\text{al*}-cs Rel Cal; (Q, R) \in cSim-cs\text{ Rel Cal}] \Rightarrow (P, R) \in cSim-cs\text{ Rel Cal}]

**lemma** weak-reduction-correspondence-simulation-impl-coupled-simulation:

**fixes** Rel :: (\text{'proc} \times \text{'proc})\text{ set}

and Cal :: (\text{'proc processCalculus})

**assumes** corrSim: weak-reduction-correspondence-simulation \( R \) \( Cal \)

**shows** weak-reduction-coupled-simulation \( \text{(cSim-cs Rel Cal)} \ Cal \)

and \( \forall P \ Q. (P, Q) \in R \rightarrow (P, Q) \in \text{cSim-cs Rel Cal} \land (Q, P) \in \text{cSim-cs Rel Cal} \)

**proof**

- **show** weak-reduction-coupled-simulation \( \text{(cSim-cs Rel Cal)} \ Cal \)

**proof** (rule allI, rule allI, rule allI, rule impI, erule conjE)
fix \( P, Q, P' \)
assume \((P, Q) \in cSim-cs \text{ Rel Cal} \) and \( P \rightarrow Cal* P' \)
thus \((\exists Q', Q \rightarrow Cal* Q' \land (P', Q') \in cSim-cs \text{ Rel Cal}) \land (\exists Q', Q \rightarrow Cal* Q' \land (Q', P') \in cSim-cs \text{ Rel Cal})\)

proof (induct arbitrary: \( P' \))
case \((left \ Q Q' P)\)
assume \((P, Q') \in \text{Rel} \) and \( P \rightarrow Cal* P' \)
with \(corrSim\) obtain \( Q''\) where \( A1: Q' \rightarrow Cal* Q'' \) and \( A2: (P', Q'') \in \text{Rel} \)
  by \(\text{blast}\)
assume \( A3: Q \rightarrow Cal* Q'\)
from this \( A1 \) have \( A4: Q \rightarrow Cal* Q''\)
  by \((\text{rule steps-add}[\text{where } P=Q \text{ and } Q=Q' \text{ and } R=Q''])\)
have \( Q'' \rightarrow Cal* Q''\)
  by \((\text{rule steps-refl})\)
with \( A2 \) have \( A5: (Q'', P') \in cSim-cs \text{ Rel Cal} \)
  by \((\text{simp add: cSim-cs.right})\)
from \( A1 A2 \) have \( (P', Q') \in cSim-cs \text{ Rel Cal} \)
  by \((\text{rule cSim-cs.left})\)
with \( A4 A5 A3 \) show \(?case\)
  by \(\text{blast}\)
next
case \((right \ P P' Q P'')\)
assume \( P \rightarrow Cal* P' \) and \( P' \rightarrow Cal* P''\)
hence \( B1: P \rightarrow Cal* P''\)
  by \((\text{rule steps-add}[\text{where } P=P \text{ and } Q=P' \text{ and } R=P''])\)
assume \( B2: (Q, P) \in \text{Rel} \)
with \(corrSim\) \( B1 \) obtain \( Q'''' P''''\) where \( B3: Q \rightarrow Cal* Q'''' \) and \( B4: P'' \rightarrow Cal* P''''\)
  and \( B5: (Q'''', P''') \in \text{Rel} \)
  by \(\text{blast}\)
from \( B4 B5 \) have \( B6: (Q''', P''') \in cSim-cs \text{ Rel Cal} \)
  by \((\text{rule cSim-cs.left})\)
have \( B7: Q \rightarrow Cal* Q''\)
  by \((\text{rule steps-refl})\)
from \( B1 B2 \) have \( (P'', Q) \in cSim-cs \text{ Rel Cal} \)
  by \((\text{rule cSim-cs.right})\)
with \( B3 B6 B7 \) show \(?case\)
  by \(\text{blast}\)
next
case \((\text{trans } P Q R P')\)
assume \( P \rightarrow Cal* P' \)
and \( \bigwedge P', P \rightarrow Cal* P' \quad (\exists Q', Q \rightarrow Cal* Q' \land (P', Q') \in cSim-cs \text{ Rel Cal}) \)
\land (\exists Q', Q \rightarrow Cal* Q' \land (Q', P') \in cSim-cs \text{ Rel Cal})\)
from this obtain \( Q1 Q2 \) where \( C1: Q \rightarrow Cal* Q1 \) and \( C2: (Q1, P') \in cSim-cs \text{ Rel Cal} \)
\land \( C3: Q \rightarrow Cal* Q2 \) and \( C4: (P', Q2) \in cSim-cs \text{ Rel Cal} \)
  by \(\text{blast}\)
assume \( C5: \bigwedge Q', Q \rightarrow Cal* Q' \quad (\exists R', R \rightarrow Cal* R' \land (Q', R') \in cSim-cs \text{ Rel Cal}) \)
\land (\exists R', R \rightarrow Cal* R' \land (R', Q') \in cSim-cs \text{ Rel Cal})\)
with \( C1 \) obtain \( R1 \) where \( C6: R \rightarrow Cal* R1 \) and \( C7: (R1, Q1) \in cSim-cs \text{ Rel Cal} \)
  by \(\text{blast}\)
from \( C7 C2 \) have \( C8: (R1, P') \in cSim-cs \text{ Rel Cal} \)
  by \((\text{rule cSim-cs.trans})\)
from \( C3 C5 \) obtain \( R2 \) where \( C9: R \rightarrow Cal* R2 \) and \( C10: (Q2, R2) \in cSim-cs \text{ Rel Cal} \)
  by \(\text{blast}\)
from \( C4 C10 \) have \( P', R2 \) \in cSim-cs \text{ Rel Cal} 
  by \((\text{rule cSim-cs.trans})\)
with \( C6 C8 C9 \) show \(?case\)
  by \(\text{blast}\)
qed
qed
next
show \( \forall P Q. (P, Q) \in \text{Rel} \rightarrow (P, Q) \in cSim-cs \text{ Rel Cal} \land (Q, P) \in cSim-cs \text{ Rel Cal} \)
proof clarify
fix P Q
have Q \rightarrow Cal* Q
by (rule steps-refl)
moreover assume (P, Q) \in Rel
ultimately show \((P, Q) \in cSim\) Rel Cal \land \((Q, P) \in cSim\) Rel Cal
by (simp add: cSim.left cSim.right)
qed
qed

lemma weak-barbed-correspondence-simulation-impl-coupled-simulation:
fixes Rel :: ('proc \times 'proc) set
and CWB :: ('proc, 'barbs) calculusWithBarbs
assumes corrSim: weak-barbed-correspondence-simulation Rel CWB
shows weak-barbed-coupled-simulation \((cSim\cdot cs\) Rel \((Calculus\ CWB))\) CWB
and \(\forall\ P\ Q.\ (P, Q) \in Rel \rightarrow (P, Q) \in cSim\cdot cs\) Rel \((Calculus\ CWB))\)
\land \((Q, P) \in cSim\cdot cs\) Rel \((Calculus\ CWB))\)
proof -
show weak-barbed-coupled-simulation \((cSim\cdot cs\) Rel \((Calculus\ CWB))\) CWB
proof from corrSim
show weak-reduction-coupled-simulation \((cSim\cdot cs\) Rel \((Calculus\ CWB))\) \((Calculus\ CWB)\)
using weak-reduction-correspondence-simulation-impl-coupled-simulation(1)[where Rel=Rel
and Cal=Calculus CWB]
by blast
next
show rel-weakly-preserves-barbs \((cSim\cdot cs\) Rel \((Calculus\ CWB))\) CWB
proof clarify
fix P Q a P'
assume \((P, Q) \in cSim\cdot cs\) Rel \((Calculus\ CWB))\) and \(P \rightarrow (Calculus\ CWB)* \ P'\) and \(P'\downarrow<\ C W B> a\)
thus \(Q\downarrow<\ C W B>a\)
proof (induct arbitrary; \(P')\)
case (left Q Q' P P')
assume \((P, Q') \in Rel\) and \(P \rightarrow (Calculus\ CWB)* \ P'\) and \(P'\downarrow<\ C W B>a\)
with corrSim obtain \(Q''\) where \(A1:\ Q' \rightarrow (Calculus\ CWB)* \ Q''\) and \(A2:\ Q''\downarrow<\ C W B>a\)
by blast
assume \(Q \rightarrow (Calculus\ CWB)* \ Q'\)
from this \(A1\) have \(Q \rightarrow (Calculus\ CWB)* \ Q''\)
by (rule steps-add)
with \(A2\) show \(Q\downarrow<\ C W B>a\)
by blast
next
case (right P P' Q P'')
assume \((Q, P) \in Rel\)
moreover assume \(P \rightarrow (Calculus\ CWB)* \ P'\) and \(P' \rightarrow (Calculus\ CWB)* \ P''\)
hence \(P \rightarrow (Calculus\ CWB)* \ P''\)
by (rule steps-add)
moreover assume \(P''\downarrow<\ C W B>a\)
ultimately show \(Q\downarrow<\ C W B>a\)
using corrSim
by blast
next
case (trans P Q R P')
assume \(\land P'.\ P \rightarrow (Calculus\ CWB)* \ P'\) \(\rightarrow P'\downarrow<\ C W B>a\ \rightarrow Q\downarrow<\ C W B>a\)
and \(P \rightarrow (Calculus\ CWB)* \ P'\) and \(P'\downarrow<\ C W B>a\)
and \(\land Q'.\ Q \rightarrow (Calculus\ CWB)* \ Q'\) \(\rightarrow Q'\downarrow<\ C W B>a\ \rightarrow R\downarrow<\ C W B>a\)
thus \(R\downarrow<\ C W B>a\)
by blast
qed
qed
qed
proof -

using weak-reduction-correspondence-simulation-impl-coupled-simulation(2)[where Rel=Rel
and Cal=Calculus CWB]

qed

lemma reduction-correspondence-simulation-condition-trans:

\[ \text{fixes } Cal \ni '\text{proc processCalculus}
\]
\[ \text{and } P \in R \ni '\text{proc}
\]
\[ \text{and Rel } \ni (\text{proc } \times \text{proc}) \text{ set}
\]
\[ \text{assumes } A1: \forall Q'. Q \rightarrow Cal* Q' \rightarrow (\exists P'' Q''). P \rightarrow Cal* P'' \land Q' \rightarrow Cal* Q'' \land (P'', Q'') \in Rel
\]
\[ \text{and } A2: \forall R'. R \rightarrow Cal* R' \rightarrow (\exists Q'' R''). Q \rightarrow Cal* Q'' \land R' \rightarrow Cal* R'' \land (Q'', R'') \in Rel
\]
\[ \text{and } A3: \text{weak-reduction-correspondence-simulation Rel Cal}
\]
\[ \text{and } A4: \text{trans Rel}
\]
\[ \text{shows } \forall R'. R \rightarrow Cal* R' \rightarrow (\exists P'' R''). P \rightarrow Cal* P'' \land R' \rightarrow Cal* R'' \land (P'', R'') \in Rel
\]

proof clarify

fix R'

assume R \rightarrow Cal* R'

with A2 obtain Q'' R'' where A5: Q \rightarrow Cal* Q'' and A6: R' \rightarrow Cal* R''

and A7: (Q'', R'') \in Rel

by blast

from A1 A5 obtain P''' Q''' where A8: P \rightarrow Cal* P''' and A9: Q'' \rightarrow Cal* Q''

and A10: (P''', Q''') \in Rel

by blast

from A3 A7 A9 obtain R''' where A11: R'' \rightarrow Cal* R''' and A12: (Q''', R''') \in Rel

by blast

from A6 A11 have A13: R' \rightarrow Cal* R'''

by (rule steps-add[where P=R' and Q=R'' and R=R'''][where Rel=Rel and Cal=Cal])

from A4 A10 A12 have (P''', R''') \in Rel

unfolding trans-def

by blast

with A8 A13 show \( \exists P'' R'' \). P \rightarrow Cal* P'' \land R' \rightarrow Cal* R'' \land (P'', R'') \in Rel

by blast

qed

The reflexive and/or transitive closure of a weak correspondence simulation is a weak correspondence simulation.
assume $P = Q$

moreover have $Q' \iff \text{Cal} \ast Q'$

by (rule steps-refl)

ultimately show $\exists P'' \ Q'', \ P \iff \text{Cal} \ast P'' \land Q' \iff \text{Cal} \ast Q'' \land (P'', \ Q'') \in \text{Rel}^+$

using A1 refl

by blast

qed

moreover

have $(P, \ Q) \in \text{Rel} \implies \exists P'' \ Q''$, $P \iff \text{Cal} \ast P'' \land Q' \iff \text{Cal} \ast Q'' \land (P'', \ Q'') \in \text{Rel}^+$

proof

- assume $(P, \ Q) \in \text{Rel}$

  with corrSim A1 obtain $P'' \ Q''$ where $P \iff \text{Cal} \ast P''$ and $Q' \iff \text{Cal} \ast Q''$

  and $(P'', \ Q'') \in \text{Rel}^+$

  by blast

thus $\exists P'' \ Q'$. $P \iff \text{Cal} \ast P'' \land Q' \iff \text{Cal} \ast Q'' \land (P'', \ Q'') \in \text{Rel}^+$

by auto

qed

ultimately show $\exists P'' \ Q''$, $P \iff \text{Cal} \ast P'' \land Q' \iff \text{Cal} \ast Q'' \land (P'', \ Q'') \in \text{Rel}^+$

by auto

qed

have $B: \ A \land \text{Rel}$. weak-reduction-correspondence-simulation $\text{Rel} \ \text{Cal}$

$\implies$ weak-reduction-correspondence-simulation $(\text{Rel}^+) \ \text{Cal}$

proof

fix $\text{Rel}$

assume weak-reduction-correspondence-simulation $\text{Rel} \ \text{Cal}$

thus weak-reduction-simulation $(\text{Rel}^+) \ \text{Cal}$

using weak-reduction-simulation-and-closures[2] where $\text{Rel}=\text{Rel}$ and $\text{Cal}=\text{Cal}$

by blast

next

fix $\text{Rel}$

assume B1: weak-reduction-correspondence-simulation $\text{Rel} \ \text{Cal}$

show $\forall P \ Q \ Q'$. $(P, \ Q) \in \text{Rel}^+ \land Q \iff \text{Cal} \ast Q'$

$\implies (\exists P'' \ Q''). P \iff \text{Cal} \ast P'' \land Q' \iff \text{Cal} \ast Q'' \land (P'', \ Q'') \in \text{Rel}^+$

proof clarify

fix $P \ Q \ Q'$

assume $(P, \ Q) \in \text{Rel}^+$ and $Q \iff \text{Cal} \ast Q'$

thus $\exists P'' \ Q''$. $P \iff \text{Cal} \ast P'' \land Q' \iff \text{Cal} \ast Q'' \land (P'', \ Q'') \in \text{Rel}^+$

proof (induct arbitrary: $Q'$)

fix $Q \ Q'$

assume $(P, \ Q) \in \text{Rel}$ and $Q \iff \text{Cal} \ast Q'$

with B1 obtain $P'' \ Q''$ where B2: $P \iff \text{Cal} \ast P''$ and B3: $Q' \iff \text{Cal} \ast Q''$

and B4: $(P'', \ Q'') \in \text{Rel}$

by blast

from B4 have $(P'', \ Q'') \in \text{Rel}^+$

by simp

with B2 B3 show $\exists P'' \ Q''$. $P \iff \text{Cal} \ast P'' \land Q' \iff \text{Cal} \ast Q'' \land (P'', \ Q'') \in \text{Rel}^+$

by blast

next

case (step $Q \ R \ R'$)

assume $\land Q'. P \iff \text{Cal} \ast Q'$

$\implies (\exists P'' \ Q''). P \iff \text{Cal} \ast P'' \land Q' \iff \text{Cal} \ast Q'' \land (P'', \ Q'') \in \text{Rel}^+$

moreover assume $(Q, \ R) \in \text{Rel}$

with B1

have $\land R' \ R \ R' \iff (\exists Q'' \ R'')$, $Q \iff \text{Cal} \ast Q'' \land R' \iff \text{Cal} \ast R'' \land (Q'', \ R'') \in \text{Rel}^+$

by blast

moreover from B1 have weak-reduction-simulation $(\text{Rel}^+) \ \text{Cal}$

using weak-reduction-simulation-and-closures[2] where $\text{Rel}=\text{Rel}$ and $\text{Cal}=\text{Cal}$

by blast

moreover have trans $(\text{Rel}^+)$

using trans-trancl[of Rel]
A weak reduction bisimulation is a relation \( R \) such that (1) if \((P, Q)\) in \( R \) and \( P \) evolves to some \( P' \) then there exists some \( Q' \) such that \( Q \) evolves to \( Q' \) and \((P', Q')\) in \( R \), and (2) if \((P, Q)\) in \( R \) and \( Q \) evolves to some \( Q' \) then there exists some \( P' \) such that \( P \) evolves to \( P' \) and \((P', Q')\) in \( R \).

**abbreviation** \( \text{weak-reduction-bisimulation} \) 
\[ :: \langle \text{proc} \times \text{proc} \rangle \Rightarrow \text{proc processCalculus} \Rightarrow \text{bool} \]

**where**
\[ \text{weak-reduction-bisimulation} \ Rel \; \text{Cal} \equiv (\forall \; P \; Q \; P'. \; (P, \; Q) \; \in \; \text{Rel} \; \land \; P \rightarrow \text{Cal} \; \Rightarrow \; (\exists \; Q'. \; Q \rightarrow \text{Cal} \; \Rightarrow \; (P', \; Q')) \in \text{Rel} \) \]

A weak barbed bisimulation is a weak reduction bisimulation that weakly respects barbs.

**abbreviation** \( \text{weak-barbed-bisimulation} \) 
\[ :: \langle \text{proc} \times \text{proc} \rangle \Rightarrow \langle \text{proc}, \text{barbs} \rangle \text{calculusWithBarbs} \Rightarrow \text{bool} \]

**where**
\[ \text{weak-barbed-bisimulation} \ Rel \; \text{CWB} \equiv \]
A symmetric weak simulation is a weak bisimulation.

**Lemma symmetric-weak-reduction-simulation-is-bisimulation:**

*Proof*

Assume $\mathsf{sym}$ $\mathcal{R}$ and $\mathsf{weak-reduction-simulation} \mathcal{R} \mathcal{C}$

Show $\mathsf{weak-reduction-bisimulation} \mathcal{R} \mathcal{C}$

By blast

**Lemma symmetric-weak-barbed-simulation-is-bisimulation:**

*Proof*

Assume $\mathsf{sym} \mathcal{R}$ and $\mathsf{weak-barbed-simulation} \mathcal{R} \mathcal{C}$

Show $\mathsf{weak-barbed-bisimulation} \mathcal{R} \mathcal{C}$

By blast

If a relation as well as its inverse are weak simulations, then this relation is a weak bisimulation.

**Lemma weak-reduction-simulations-impl-bisimulation:**

*Proof*

Assume $\mathsf{sim} : \mathsf{weak-reduction-simulation} \mathcal{R} \mathcal{C}$ and $\mathsf{sim}^{-1} : \mathsf{weak-reduction-simulation} (\mathcal{R}^{-1}) \mathcal{C}$

Show $\mathsf{weak-reduction-bisimulation} \mathcal{R} \mathcal{C}$

By auto

**Lemma weak-reduction-bisimulations-impl-inverse-is-simulation:**

*Proof*

Assume $\mathsf{bisim} : \mathsf{weak-reduction-bisimulation} \mathcal{R} \mathcal{C}$

Show $\mathsf{weak-reduction-simulation} (\mathcal{R}^{-1}) \mathcal{C}$

By clarify
from A2 have \((P', Q') \in \text{Rel}^{-1}\)
  by simp

with A1 show \(\exists Q'. \ Q \longrightarrow \text{Cal}^* \{Q' \land (P', Q') \in \text{Rel}^{-1}\}\)
  by blast

qed

lemma weak-reduction-simulations-iff-bisimulation:

fixes Rel :: ('proc 'proc) set
  and Cal :: 'proc processCalculus
shows (weak-reduction-simulation Rel Cal \land weak-reduction-simulation (\text{Rel}^{-1}) Cal)
  = weak-reduction-bisimulation Rel Cal

using weak-reduction-simulations-impl-bisimulation[where Rel=\text{Rel} and Cal=\text{Cal}]
weak-reduction-bisimulations-impl-inverse-is-simulation[where Rel=\text{Rel} and Cal=\text{Cal}]

by blast

lemma weak-barbed-simulations-iff-bisimulation:

fixes Rel :: ('proc 'proc) set
  and CWB :: ('proc, 'barbs) calculusWithBarbs
shows (weak-barbed-simulation Rel CWB \land weak-barbed-simulation (\text{Rel}^{-1}) CWB)
  = weak-barbed-bisimulation Rel CWB

proof (rule iffI,erule conjE)
  assume sim: weak-barbed-simulation Rel CWB
  and rev: weak-barbed-simulation (\text{Rel}^{-1}) CWB
  hence weak-reduction-bisimulation Rel (\text{Calculus CWB})
    using weak-reduction-simulations-impl-bisimulation[where Rel=\text{Rel} and Cal=\text{Cal}]

  by blast

moreover from sim have rel-weakly-preserves-barbs Rel CWB
  by simp

moreover from rev have rel-weakly-reflects-barbs Rel CWB
  by simp

ultimately show weak-barbed-bisimulation Rel CWB
  by blast

next
  assume bisim: weak-barbed-bisimulation Rel CWB
  hence weak-barbed-simulation Rel CWB
  by simp

moreover from bisim have weak-reduction-simulation (\text{Rel}^{-1}) (\text{Calculus CWB})
    using weak-reduction-bisimulations-impl-inverse-is-simulation[where Rel=\text{Rel}]

  by simp

moreover from bisim have rel-weakly-reflects-barbs Rel CWB
  by blast

hence rel-weakly-preserves-barbs (\text{Rel}^{-1}) CWB
  by simp

ultimately show weak-barbed-simulation Rel CWB \land weak-barbed-simulation (\text{Rel}^{-1}) CWB
  by blast

qed

A weak bisimulation is a weak correspondence simulation.

lemma weak-reduction-bisimulation-is-correspondence-simulation:

fixes Rel :: ('proc 'proc) set
  and Cal :: 'proc processCalculus
assumes bisim: weak-reduction-bisimulation Rel Cal
shows weak-reduction-correspondence-simulation Rel Cal

proof
  from bisim show weak-reduction-simulation Rel Cal
  by blast

next
  show \(\forall P Q Q'. \ (P, Q) \in \text{Rel} \land Q \longrightarrow \text{Cal}^* \{Q' \land (P', Q') \in \text{Rel}\}\)

  \longrightarrow (\exists P'' Q''. \ P' \longrightarrow \text{Cal}^* \{P'' \land Q'' \longrightarrow \text{Cal}^* \{Q'' \land (P'', Q'') \in \text{Rel}\}\}

proof clarify
  fix P Q Q'


The reflexive, symmetric, and/or transitive closure of a weak bisimulation is a weak bisimulation.

**lemma** weak-barbed-bisimulation-is-correspondence-simulation:

- **fixes** \( \text{Rel} :: (\text{'proc} \times \text{'proc}) \text{ set} \)
- **and** \( \text{CWB} :: (\text{'proc}, \text{ 'barbs}) \text{ calculusWithBarbs} \)
- **assumes** \( \text{bisim} :: \text{weak-barbed-bisimulation} \text{ Rel} \text{ CWB} \)
- **shows** weak-barbed-correspondence-simulation \( \text{Rel} \text{ CWB} \)
- **using** \( \text{bisim} :: \text{weak-barbed-bisimulation} \text{ Rel} \text{ CWB} \)
  - **by** blast

**proof**

- **from** \( \text{bisim} \text{ show} \ A :: \text{weak-reduction-bisimulation} \ (\text{Rel}^+) \text{ Cal} \)
  - **by** (auto simp add: refl, blast+)
- **have** \( B :: \mathbb{A} \text{Rel} \text{. weak-reduction-bisimulation} \text{ Rel} \text{ Cal} \)
  - **implies** weak-reduction-bisimulation \( \text{symcl} \text{ Rel} \text{ Cal} \)
  - **by** (auto simp add: symcl_def, blast+)
- **from** \( \text{bisim} \text{ B[where Rel=Rel]} \text{ show} \text{ weak-reduction-bisimulation} \ (\text{symcl} \text{ Rel}) \text{ Cal} \)
  - **by** blast
- **have** \( C :: \mathbb{A} \text{Rel} \text{. weak-reduction-bisimulation} \text{ Rel} \text{ Cal} \)
  - **implies** weak-reduction-bisimulation \( \text{Rel}^+ \text{ Cal} \)

**proof**

- **fix** \( \text{Rel} \)
- **assume** \( \text{weak-reduction-bisimulation} \text{ Rel} \text{ Cal} \)
- **thus** weak-reduction-simulation \( \text{(Rel}^+) \text{ Cal} \)
  - **using** weak-reduction-simulation-and-closures[2][where Rel=Rel and Cal=Cal]
  - **by** blast

**next**

- **fix** \( \text{Rel} \)
- **assume** \( C1 :: \text{weak-reduction-bisimulation} \text{ Rel} \text{ Cal} \)
- **show** \( \forall \ P \ Q \ Q'. \ (P, Q) \in \text{Rel}^+ \land Q \rightarrow \text{Cal}^* \ Q' \rightarrow \exists P'. \ P \rightarrow \text{Cal}^* \ P' \land (P', Q') \in \text{Rel}^+ \)

**proof**

- **clarify**
- **fix** \( P \ Q \ Q' \)
- **assume** \( (P, Q) \in \text{Rel}^+ \text{ and } Q \rightarrow \text{Cal}^* \ Q' \)
- **thus** \( \exists P'. \ P \rightarrow \text{Cal}^* \ P' \land (P', Q') \in \text{Rel}^+ \)
- **proof** (induct arbitrary: \( Q' \))
  - **fix** \( Q \ Q' \)
  - **assume** \( (P, Q) \in \text{Rel} \text{ and } Q \rightarrow \text{Cal}^* \ Q' \)
  - **thus** \( \exists P'. \ P \rightarrow \text{Cal}^* \ P' \land (P', Q') \in \text{Rel}^+ \)
  - **with** \( C1 \text{ obtain} \ P' :: \text{where P \rightarrow Cal}^* \ P' \text{ and (P', Q')} \in \text{Rel} \)
    - **by** blast
  - **thus** \( \exists P'. \ P \rightarrow \text{Cal}^* \ P' \land (P', Q') \in \text{Rel}^+ \)
by auto
next
case (step Q R R')
  assume (Q, R) ∈ Rel and R →→ Cal R'
  with C1 obtain Q' where C2: Q →→ Cal Q' and C3: (Q', R') ∈ Rel'
  by blast
  assume Q'. Q →→ Cal Q' ⇒ ∃ P'. P →→ Cal P' ∧ (P', Q') ∈ Rel'
  with C2 obtain P' where C4: P →→ Cal P' and C5: (P', Q') ∈ Rel'
  by blast
  from C5 C3 have (P', R') ∈ Rel'
  by simp
  with C4 show ∃ P'. P →→ Cal P' ∧ (P', R') ∈ Rel'
  by blast
qed
qed
qed
from bisim C[where Rel=Rel] show weak-reduction-bisimulation (Rel') Cal
  by blast
from A B[where Rel=Rel'] show weak-reduction-bisimulation (symcl (Rel')) Cal
  by blast
from A C[where Rel=Rel'] show weak-reduction-bisimulation (Rel') Cal
  using trancl-refcl[of Rel]
  by auto
from A B[where Rel=Rel'] C[where Rel=symcl (Rel')]
  show weak-reduction-bisimulation ((symcl (Rel'))') Cal
  by blast
qed

lemma weak-barbed-bisimulation-and-closures:
  fixes Rel :: ('proc × 'proc) set
  and CWB :: ('proc, 'barbs) calculusWithBarbs
  assumes bisim: weak-barbed-bisimulation Rel CWB
  shows weak-barbed-bisimulation (Rel') CWB
  and weak-barbed-bisimulation (symcl Rel) CWB
  and weak-barbed-bisimulation (Rel') CWB
  and weak-barbed-bisimulation (symcl (Rel')) CWB
  and weak-barbed-bisimulation ((symcl (Rel'))') CWB
proof
  from bisim show weak-barbed-bisimulation (Rel') CWB
    using weak-reduction-bisimulation-and-closures(1)[where Rel=Rel and Cal=Calculus CWB]
      weak-respection-of-barbs-and-closures(1)[where Rel=Rel and CWB=CWB]
    by fast
next
  from bisim show weak-barbed-bisimulation (symcl Rel) CWB
    using weak-reduction-bisimulation-and-closures(2)[where Rel=Rel and Cal=Calculus CWB]
      weak-respection-of-barbs-and-closures(2)[where Rel=Rel and CWB=CWB]
    by blast
next
  from bisim show weak-barbed-bisimulation (Rel') CWB
    using weak-reduction-bisimulation-and-closures(3)[where Rel=Rel and Cal=Calculus CWB]
      weak-respection-of-barbs-and-closures(3)[where Rel=Rel and CWB=CWB]
    by blast
next
  from bisim show weak-barbed-bisimulation (symcl (Rel')) CWB
    using weak-reduction-bisimulation-and-closures(4)[where Rel=Rel and Cal=Calculus CWB]
      weak-respection-of-barbs-and-closures(4)[where Rel=Rel and CWB=CWB]
    by blast
next
  from bisim show weak-barbed-bisimulation (Rel') CWB
    using weak-reduction-bisimulation-and-closures(5)[where Rel=Rel and Cal=Calculus CWB]
weak-respection-of-barbs-and-closures(5)[where \(\text{Rel} = \text{Rel} \land \text{CWB} = \text{CWB}\)]

by blast

next

from bisim show weak-barbed-bisimulation ((symcl (\(\text{Rel}^{-1}\)))^+\) \(\text{CWB}\)

using weak-reduction-bisimulation-and-closures(6)[where \(\text{Rel} = \text{Rel} \land \text{Cal} = \text{Calculus} \text{CWB}\)]

weak-respection-of-barbs-and-closures(6)[where \(\text{Rel} = \text{Rel} \land \text{CWB} = \text{CWB}\)]

by blast

qed

A strong reduction bisimulation is relation R such that (1) if \((P, Q) \in R\) and \(P'\) is a derivative of \(P\) then there exists some \(Q'\) such that \(Q'\) is a derivative of \(Q\) and \((P', Q') \in R\), and (2) if \((P, Q) \in R\) and \(Q'\) is a derivative of \(Q\) then there exists some \(P'\) such that \(P'\) is a derivative of \(P\) and \((P', Q') \in R\).

abbreviation strong-reduction-bisimulation
:: ('proc \times 'proc) set ⇒ 'proc processCalculus ⇒ bool

where

strong-reduction-bisimulation \(\text{Rel} \text{Cal} \equiv \)

\((\forall P Q P'. (P, Q) \in \text{Rel} \land P \rightarrow\text{Cal} P' \rightarrow (\exists Q'. Q \rightarrow\text{Cal} Q' \land (P', Q') \in \text{Rel})) \land (\forall P Q Q'. (P, Q) \in \text{Rel} \land Q \rightarrow\text{Cal} Q' \rightarrow (\exists P'. P \rightarrow\text{Cal} P' \land (P', Q') \in \text{Rel}))\)

A strong barbed bisimulation is strong reduction bisimulation that respects barbs.

abbreviation strong-barbed-bisimulation
:: ('proc \times 'proc) set ⇒ ('proc, 'barbs) calculusWithBarbs ⇒ bool

where

strong-barbed-bisimulation \(\text{Rel} \text{CWB} \equiv \)

strong-reduction-bisimulation \(\text{Rel} \text{Calculus} \text{CWB}) \land \text{rel-respects-barbs} \text{Rel} \text{CWB}\)

A symetric strong simulation is a strong bisimulation.

lemma symm-strong-reduction-simulation-is-bisimulation:

fixes \(\text{Rel} \:: \ ('proc \times 'proc) \text{set} \)

and \(\text{Cal} :: \ 'proc \text{processCalculus} \)

assumes sym \(\text{Rel} \)

and strong-reduction-simulation \(\text{Rel} \text{Cal} \)

shows strong-reduction-bisimulation \(\text{Rel} \text{Cal} \)

using assms symD[of \text{Rel}] \[\text{of \text{Rel}}\]

by blast

lemma symm-strong-barbed-simulation-is-bisimulation:

fixes \(\text{Rel} :: \ ('proc \times 'proc) \text{set} \)

and \(\text{CWB} :: \ ('proc, 'barbs) \text{calculusWithBarbs} \)

assumes sym \(\text{Rel} \)

and strong-barbed-simulation \(\text{Rel} \text{CWB} \)

shows strong-barbed-bisimulation \(\text{Rel} \text{CWB} \)

using assms symD[of \text{Rel}] \[\text{of \text{Rel}}\]

by blast

If a relation as well as its inverse are strong simulations, then this relation is a strong bisimulation.

lemma strong-reduction-simulations-impl-bisimulation:

fixes \(\text{Rel} :: \ ('proc \times 'proc) \text{set} \)

and \(\text{Cal} :: \ 'proc \text{processCalculus} \)

assumes sim:: strong-reduction-simulation \(\text{Rel} \text{Cal} \)

and simInv:: strong-reduction-simulation \((\text{Rel}^{-1}) \) \text{Cal} \)

shows strong-reduction-bisimulation \(\text{Rel} \text{Cal} \)

proof auto

fix \(P Q P'\)

assume \((P, Q) \in \text{Rel} \land P \rightarrow\text{Cal} P'\)

with sim show \(\exists Q'. Q \rightarrow\text{Cal} Q' \land (P', Q') \in \text{Rel} \)

by simp

next
fix \( P \ Q \ Q' \)
assume \((P, Q) \in \text{Rel}\)
hence \((Q, P) \in \text{Rel}^{-1}\)
by simp
moreover assume \(Q \mapsto \text{Cal} Q'\)
ultimately obtain \(P'\) where \(A1: P \mapsto \text{Cal} P'\) and \(A2: (Q', P') \in \text{Rel}^{-1}\)
using simInv
by blast
from \(A2\) have \((P', Q') \in \text{Rel}\)
by induct
with \(A1\) show \(\exists P'. P \mapsto \text{Cal} P' \land (P', Q') \in \text{Rel}^{-1}\)
by blast
qed

lemma strong-reduction-bisimulations-impl-inverse-is-simulation:
fixes \(\text{Rel}:: ('\text{proc} \times ' \text{proc}) \text{ set}\)
and \(\text{Cal}:: '\text{proc} \text{ processCalculus}\)
assumes bisim: strong-reduction-bisimulation \(\text{Rel} \ \text{Cal}\)
shows strong-reduction-simulation \((\text{Rel}^{-1}) \ \text{Cal}\)
proof clarify
fix \(P \ Q \ P'\)
assume \((Q, P) \in \text{Rel}\)
moreover assume \(P \mapsto \text{Cal} P'\)
ultimately obtain \(Q'\) where \(A1: Q \mapsto \text{Cal} Q'\) and \(A2: (Q', P') \in \text{Rel}^{-1}\)
using bisim
by blast
from \(A2\) have \((P', Q') \in \text{Rel}^{-1}\)
by simp
with \(A1\) show \(\exists Q'. Q \mapsto \text{Cal} Q' \land (P', Q') \in \text{Rel}^{-1}\)
by blast
qed

lemma strong-reduction-simulations-iff-bisimulation:
fixes \(\text{Rel}:: ('\text{proc} \times ' \text{proc}) \text{ set}\)
and \(\text{CWB}:: ('\text{proc}, '\text{barbs}) \text{ calculusWithBarbs}\)
shows \((\text{strong-reduction-simulation} \ \text{Rel} \ \text{CWB} \land \text{strong-reduction-simulation} \ ((\text{Rel}^{-1}) \ \text{CWB}))\)
= strong-reduction-bisimulation \(\text{Rel} \ \text{CWB}\)
using strong-reduction-simulations-impl-bisimulation[where \(\text{Rel}=\text{Rel}\) and \(\text{Cal}=\text{Cal}\)]
strong-reduction-bisimulations-impl-inverse-is-simulation[where \(\text{Rel}=\text{Rel}\)]
by blast

lemma strong-barbed-simulations-iff-bisimulation:
fixes \(\text{Rel}:: ('\text{proc} \times ' \text{proc}) \text{ set}\)
and \(\text{CWB}:: ('\text{proc}, '\text{barbs}) \text{ calculusWithBarbs}\)
shows \((\text{strong-barbed-simulation} \ \text{Rel} \ \text{CWB} \land \text{strong-barbed-simulation} \ ((\text{Rel}^{-1}) \ \text{CWB}))\)
= strong-barbed-bisimulation \(\text{Rel} \ \text{CWB}\)
proof (rule iffI, erule conjE)
assume sim: strong-barbed-simulation \(\text{Rel} \ \text{CWB}\)
and rev: strong-barbed-simulation \((\text{Rel}^{-1}) \ \text{CWB}\)
hence strong-reduction-bisimulation \(\text{Rel} \ (\text{Calculus} \ \text{CWB})\)
using strong-reduction-simulations-impl-bisimulation[where \(\text{Rel}=\text{Rel}\) and \(\text{Cal}=\text{Calculus} \ \text{CWB}\)]
by blast
moreover from sim have rel-preserves-barbs \(\text{Rel} \ \text{CWB}\)
by simp
moreover from rev have rel-reflects-barbs \(\text{Rel} \ \text{CWB}\)
by simp
ultimately show strong-barbed-bisimulation \(\text{Rel} \ \text{CWB}\)
by blast
next
assume bisim: strong-barbed-bisimulation \(\text{Rel} \ \text{CWB}\)
hence strong-barbed-simulation \(\text{Rel} \ \text{CWB}\)
A strong bisimulation is a weak bisimulation.

**Lemma** \textit{strong-impl-weak-reduction-bisimulation}:

- **Fixes** \(\text{Rel} :: (\text{proc} \times \text{proc}) \set)
- **And** \(\text{Cal} :: \text{\text{proc processCalculus}}\)
- **Assumes** \text{bisim} : \text{strong-reduction-bisimulation} \(\text{Rel} \Cal\)
- **Shows** \text{weak-reduction-bisimulation} \(\text{Rel} \Cal\)

**Proof**

- From \text{bisim} show \text{weak-reduction-simulation} \(\text{Rel} \Cal\)
- Using \text{strong-impl-weak-reduction-simulation}[\text{where}\ \text{Rel}=\text{Rel}]
  - By \text{blast}

**Next**

- Show \(\forall P \ Q \ Q'. (P, Q) \in \text{Rel} \wedge Q \rightarrow \text{Cal}^* Q' \rightarrow (\exists P'. P \rightarrow \text{Cal}^* P' \wedge (P', Q') \in \text{Rel})\)

**Proof**

- Clarify
  - Fix \(P \ Q \ Q'\)
  - Assume \(A_1: (P, Q) \in \text{Rel}\)
  - Assume \(Q \rightarrow \text{Cal}^* Q'\)
  - From this obtain \(n\) where \(Q \rightarrow \text{Cal}^n Q'\)
    - By (auto simp add: steps-def)
  - Thus \(\exists P'. P \rightarrow \text{Cal}^* P' \wedge (P', Q') \in \text{Rel}\)

**Proof**

- (induct \(n\) arbitrary: \(Q'\))
  - Case 0
    - Assume \(Q \rightarrow \text{Cal}^0 Q'\)
    - Hence \(Q = Q'\)
    - By (simp add: steps-refl)
    - Moreover have \(P \rightarrow \text{Cal}^* P\)
      - By (rule steps-refl)
  - Ultimately show \(\exists P'. P \rightarrow \text{Cal}^* P' \wedge (P', Q') \in \text{Rel}\)
    - Using \(A_1\)
      - By blast

**Next**

- Case \(\text{Suc} \ n \ Q''\)
  - Assume \(Q \rightarrow \text{Cal} \text{Suc} \ n \ Q''\)
  - From this obtain \(Q'\) where \(A_2: Q \rightarrow \text{Cal}^n Q'\) and \(A_3: Q' \rightarrow \text{Cal} Q''\)
    - By auto
  - Assume \(\bigwedge Q'. Q \rightarrow \text{Cal}^n Q' \rightarrow \exists P'. P \rightarrow \text{Cal}^* P' \wedge (P', Q') \in \text{Rel}\)
  - With \(A_2\) obtain \(P'\) where \(A_4: P \rightarrow \text{Cal}^* P'\) and \(A_5: (P', Q') \in \text{Rel}\)
    - By blast
  - From \text{bisim} \(A_5 \ A_3\) obtain \(P''\) where \(A_6: P' \rightarrow \text{Cal} P''\) and \(A_7: (P'', Q'') \in \text{Rel}\)
    - By blast
  - From \(A_4 \ A_6\) have \(P \rightarrow \text{Cal}^* P''\)
    - Using \text{steps-add}[\text{where}\ P=P' \text{ and } Q-Q' \text{ and } R=R'']
      - By (simp add: step-to-steps)
  - With \(A_7\) show \(\exists P'. P \rightarrow \text{Cal}^* P' \wedge (P', Q'') \in \text{Rel}\)
    - By blast

qed
lemma strong-barbed-bisimulation-impl-weak-respection-of-barbs:
  fixes Rel :: ('proc × 'proc) set
  and CWB :: ('proc, 'barbs) calculusWithBarbs
  assumes bisim: strong-barbed-bisimulation Rel CWB
  shows rel-weakly-respects-barbs Rel CWB
proof
  from bisim show rel-weakly-preserves-barbs Rel CWB
    using strong-barbed-simulation-impl-weak-preservation-of-barbs
      [where Rel=Rel and CWB=CWB]
    by blast
next
  show rel-weakly-reflects-barbs Rel CWB
  proof
    clarify
    fix P Q a Q'
    assume (P, Q) ∈ Rel and Q ↦−→ (Calculus CWB)* Q'
    with bisim obtain P' where A1: P ↦−→ (Calculus CWB)* P' and A2: (P', Q') ∈ Rel
      using strong-impl-weak-reduction-bisimulation
        [where Rel=Rel and Cal=Calculus CWB]
    by blast
    assume Q'↓<CWB>a
    with bisim A2 have P'↓<CWB>a
      by blast
    with A1 show P⇓<CWB>a
      by blast
  qed
qed

lemma strong-impl-weak-barbed-bisimulation:
  fixes Rel :: ('proc × 'proc) set
  and CWB :: ('proc, 'barbs) calculusWithBarbs
  assumes bisim: strong-barbed-bisimulation Rel CWB
  shows weak-barbed-bisimulation Rel CWB
    using bisim
      strong-impl-weak-reduction-bisimulation
        [where Rel=Rel and Cal=Calculus CWB]
      strong-barbed-bisimulation-impl-weak-respection-of-barbs
        [where Rel=Rel and CWB=CWB]
    by blast

The reflexive, symmetric, and/or transitive closure of a strong bisimulation is a strong bisimulation.

lemma strong-reduction-bisimulation-and-closures:
  fixes Rel :: ('proc × 'proc) set
  and Cal :: 'proc processCalculus
  assumes bisim: strong-reduction-bisimulation Rel Cal
  shows strong-reduction-bisimulation (Rel=) Cal
    and strong-reduction-bisimulation (symcl Rel) Cal
    and strong-reduction-bisimulation (Rel+) Cal
    and strong-reduction-bisimulation (symcl (Rel=)) Cal
    and strong-reduction-bisimulation (symcl (Rel=)+) Cal
    and strong-reduction-bisimulation (symcl (Rel=))+ Cal
proof
  from bisim show A: strong-reduction-bisimulation (Rel=) Cal
    by (auto simp add: refl, blast+)
  have B: ∧ Rel. strong-reduction-bisimulation Rel Cal
    → strong-reduction-bisimulation (symcl Rel) Cal
    by (auto simp add: symcl-def, blast+)
  from bisim B [where Rel=Rel] show strong-reduction-bisimulation (symcl Rel) Cal
    by blast
  have C: ∧ Rel. strong-reduction-bisimulation Rel Cal
    → strong-reduction-bisimulation (Rel+) Cal
proof
  fix Rel
  assume strong-reduction-bisimulation Rel Cal
  thus strong-reduction-simulation (Rel+) Cal
using strong-reduction-simulation-and-closures\[2\]\[where\] Rel=Rel and Cal=Cal

by blast

next

fix Rel

assume C1: strong-reduction-bisimulation Rel Cal

show \( \forall P Q Q'. (P, Q) \in \text{Rel}^+ \rightarrow ( \exists P'. P \rightarrow \text{Cal} P' \wedge (P', Q') \in \text{Rel}^+) \)

proof clarify

\[\begin{aligned}
\text{fix } P \text{ Q} Q' \\
\text{assume } (P, Q) \in \text{Rel}^+ \text{ and } Q \rightarrow \text{Cal} Q' \\
\text{thus } \exists P'. P \rightarrow \text{Cal} P' \wedge (P', Q') \in \text{Rel}^+
\end{aligned}\]

by blast

next

\[\begin{aligned}
\text{case } (\text{step } Q R R') \\
\text{assume } (Q, R) \in \text{Rel}^+ \text{ and } R \rightarrow \text{Cal} R' \\
\text{with } C1 \text{ obtain } P' \text{ where } P \rightarrow \text{Cal} P' \wedge (P', Q') \in \text{Rel}^+
\end{aligned}\]

by blast

ass \( \wedge Q'. Q \rightarrow \text{Cal} Q' \rightarrow ( \exists P'. P \rightarrow \text{Cal} P' \wedge (P', Q') \in \text{Rel}^+) \)

by blast

\[\begin{aligned}
\text{with } C2 \text{ obtain } P' \text{ where } C4: P \rightarrow \text{Cal} P' \wedge C5: (P', Q') \in \text{Rel}^+
\end{aligned}\]

by blast

\[\begin{aligned}
\text{next } C5 \text{ C3 have } (P', R') \in \text{Rel}^+ \\
\text{by simp} \\
\text{with } C4 \text{ show } \exists P'. P \rightarrow \text{Cal} P' \wedge (P', R') \in \text{Rel}^+
\end{aligned}\]

by blast

\[\begin{aligned}
\text{qed} \\
\text{qed} \\
\text{qed}
\end{aligned}\]

\[\begin{aligned}
\text{from } \text{bisim } C[\text{where } \text{Rel}=\text{Rel}] \text{ show } \text{strong-reduction-bisimulation } (\text{Rel}^+) \text{ Cal}
\end{aligned}\]

by blast

from A B[\text{where } \text{Rel}^=\text{Rel}^] 

show \text{strong-reduction-bisimulation } (\text{symcl } (\text{Rel}^=)) \text{ Cal}

by blast

from A C[\text{where } \text{Rel}^=\text{Rel}^] 

show \text{strong-reduction-bisimulation } (\text{Rel}^+) \text{ Cal}

using trancl-refcl[\text{of } \text{Rel}]

by auto

from A B[\text{where } \text{Rel}^=\text{Rel}^] C[\text{where } \text{Rel}^=\text{symcl } (\text{Rel}^=)]

show \text{strong-reduction-bisimulation } (\text{symcl } (\text{Rel}^=)^+) \text{ Cal}

by blast

\[\begin{aligned}
\text{qed}
\end{aligned}\]

\[\begin{aligned}
\text{lemma } \text{strong-barbed-bisimulation-and-closures}:
\end{aligned}\]

\[\begin{aligned}
\text{fixes } \text{Rel} :: '\text{proc} \times '\text{proc} \text{ set}
\text{and } \text{CWB} :: '\text{proc} \times '\text{barbs} \text{ calculusWithBarbs}
\text{assumes } \text{bisim} : \text{strong-barbed-bisimulation } \text{Rel } \text{CWB}
\text{shows } \text{strong-barbed-bisimulation } (\text{Rel}^=) \text{ CWB}
\text{and } \text{strong-barbed-bisimulation } (\text{symcl } \text{Rel}) \text{ CWB}
\text{and } \text{strong-barbed-bisimulation } (\text{Rel}^+) \text{ CWB}
\text{and } \text{strong-barbed-bisimulation } (\text{symcl } (\text{Rel}^=)) \text{ CWB}
\text{and } \text{strong-barbed-bisimulation } (\text{Rel}^+) \text{ CWB}
\text{and } \text{strong-barbed-bisimulation } ((\text{symcl } (\text{Rel}^=))^+) \text{ CWB}
\text{proof –}
\end{aligned}\]

\[\begin{aligned}
\text{from } \text{bisim } \text{show } \text{strong-barbed-bisimulation } (\text{Rel}^=) \text{ CWB}
\end{aligned}\]

using strong-reduction-bisimulation-and-closures[1][\text{where } \text{Rel}=\text{Rel} \text{ and } \text{Cal}=\text{Calculus} \text{ CWB}]

respection-of-barbs-and-closures[1][\text{where } \text{Rel}=\text{Rel} \text{ and } \text{CWB}=\text{CWB}]

\[\begin{aligned}
\text{respection-of-barbs-and-closures[1][\text{where } \text{Rel}=\text{Rel} \text{ and } \text{CWB}=\text{CWB}]}
\end{aligned}\]
by fast
next
from bisim show strong-barbed-bisimulation (symcl Rel) CWB
using strong-reduction-bisimulation-and-closures(2)[where Rel=Rel and Cal=Calculus CWB]
  respection-of-barbs-and-closures(2)[where Rel=Rel and CWB=CWB]
  by blast
next
from bisim show strong-barbed-bisimulation (Rel) CWB
using strong-reduction-bisimulation-and-closures(3)[where Rel=Rel and Cal=Calculus CWB]
  respection-of-barbs-and-closures(3)[where Rel=Rel and CWB=CWB]
  by blast
next
from bisim show strong-barbed-bisimulation (symcl (Rel)) CWB
using strong-reduction-bisimulation-and-closures(4)[where Rel=Rel and Cal=Calculus CWB]
  respection-of-barbs-and-closures(4)[where Rel=Rel and CWB=CWB]
  by blast
next
from bisim show strong-barbed-bisimulation (Rel) CWB
using strong-reduction-bisimulation-and-closures(5)[where Rel=Rel and Cal=Calculus CWB]
  respection-of-barbs-and-closures(5)[where Rel=Rel and CWB=CWB]
  by blast
next
from bisim show strong-barbed-bisimulation ((symcl (Rel))*) CWB
using strong-reduction-bisimulation-and-closures(6)[where Rel=Rel and Cal=Calculus CWB]
  respection-of-barbs-and-closures(6)[where Rel=Rel and CWB=CWB]
  by blast
qed

3.6 Step Closure of Relations

The step closure of a relation on process terms is the transitive closure of the union of the relation and the inverse of the reduction relation of the respective calculus.

inductive-set stepsClosure :: ('a × 'a) set ⇒ 'a processCalculus ⇒ ('a × 'a) set
  for Rel :: ('a × 'a) set
  and Cal :: 'a processCalculus
  where
  rel: (P, Q) ∈ Rel ⇒ (P, Q) ∈ stepsClosure Rel Cal
  steps: P →→ Cal⇒ P' ⇒ (P', P) ∈ stepsClosure Rel Cal
  trans: [(P, Q) ∈ stepsClosure Rel Cal; (Q, R) ∈ stepsClosure Rel Cal]
  ⇒ (P, R) ∈ stepsClosure Rel Cal

abbreviation stepsClosureInfix ::
  'a ⇒ ('a × 'a) set ⇒ 'a processCalculus ⇒ 'a ⇒ bool (- R→<,→> - [75, 75, 75, 75] 80)
  where
  P R→<Rel,Cal> Q ⇔ (P, Q) ∈ stepsClosure Rel Cal

Applying the steps closure twice does not change the relation.

lemma steps-closure-of-steps-closure:
  fixes Rel :: ('a × 'a) set
  and Cal :: 'a processCalculus
  shows stepsClosure (stepsClosure Rel Cal) Cal = stepsClosure Rel Cal
proof auto
  fix P Q
  assume P R→<stepsClosure Rel Cal,Cal> Q
  thus P R→<Rel,Cal> Q
proof induct
  case (rel P Q)
  assume P R→<Rel,Cal> Q
  thus P R→<Rel,Cal> Q
  by simp

40
next
case (steps P P')
assume P R→< Rel, Cal > P'
thus P' R→< Rel, Cal > P
  by (rule stepsClosure.steps)
next
case (trans P Q R)
assume P R→< Rel, Cal > Q and Q R→< Rel, Cal > R
thus P R→< Rel, Cal > R
  by (rule stepsClosure.trans)
qed
next
fix P Q
assume P R→< Rel, Cal > Q and Q R→< Rel, Cal > R
thus P R→< stepsClosure Rel Cal Cal > Q
  by (rule stepsClosure.rel)
qed

The steps closure is a preorder.

lemma stepsClosure-refl:
  fixes Rel :: ('a × 'a) set
  and Cal :: 'a processCalculus
  shows refl (stepsClosure Rel Cal)
unfolding refl-on-def
proof auto
  fix P
  have P R→ Cal* P
    by (rule steps-refl)
  thus P R→< Rel, Cal > P
    by (rule stepsClosure.steps)
qed

lemma refl-trans-closure-of-rel-impl-steps-closure:
  fixes Rel :: ('a × 'a) set
  and Cal :: 'a processCalculus
  and P Q :: 'a
  assumes (P, Q) ∈ Rel* shows P R→< Rel, Cal > Q
using assms
proof induct
  show P R→< Rel, Cal > Q
    using stepsClosure-refl[of Rel Cal]
      unfolding refl-on-def by simp
next
  case (step Q R)
  assume (Q, R) ∈ Rel and P R→< Rel, Cal > Q
  thus P R→< Rel, Cal > R
    using stepsClosure.rel[of Q R Rel Cal] stepsClosure.trans[of P Q Rel Cal R]
      by blast
qed

The steps closure of a relation is always a weak reduction simulation.

lemma steps-closure-is-weak-reduction-simulation:
  fixes Rel :: ('a × 'a) set
  and Cal :: 'a processCalculus
  shows weak-reduction-simulation (stepsClosure Rel Cal) Cal
proof clarify
  fix P Q P'
  assume P R→< Rel, Cal > Q and P R→ Cal* P'

thus $\exists Q'. \ P \mapsto Cal* \ Q' \land P' \mapsto<Rel,Cal> \ Q'$

proof (induct arbitrary: $P'$)

<table>
<thead>
<tr>
<th>case (rel $P Q$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>assume $P \mapsto Cal* \ P'$</td>
</tr>
<tr>
<td>hence $P' \mapsto&lt;Rel,Cal&gt; \ P$</td>
</tr>
<tr>
<td>by (rule stepsClosure.steps)</td>
</tr>
<tr>
<td>moreover assume $(P, Q) \in Rel$</td>
</tr>
<tr>
<td>hence $P \mapsto&lt;Rel,Cal&gt; \ Q$</td>
</tr>
<tr>
<td>by (simp add: stepsClosure.rel)</td>
</tr>
<tr>
<td>ultimately have $P' \mapsto&lt;Rel,Cal&gt; \ Q'$</td>
</tr>
<tr>
<td>by (rule stepsClosure.trans)</td>
</tr>
<tr>
<td>thus $\exists Q'. \ P \mapsto Cal* \ Q' \land P' \mapsto&lt;Rel,Cal&gt; \ Q'$</td>
</tr>
<tr>
<td>using steps-refl[where $Cal=Cal$ and $P=Q$]</td>
</tr>
<tr>
<td>by blast</td>
</tr>
</tbody>
</table>

next

<table>
<thead>
<tr>
<th>case (steps $P P' P''$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>assume $P \mapsto Cal* \ P'$ and $P' \mapsto Cal* \ P''$</td>
</tr>
<tr>
<td>hence $P \mapsto Cal* \ P''$</td>
</tr>
<tr>
<td>by (rule steps-add)</td>
</tr>
<tr>
<td>moreover have $P'' \mapsto&lt;Rel,Cal&gt; \ P''$</td>
</tr>
<tr>
<td>using stepsClosure-refl[where $Rel=Rel$ and $Cal=Cal$]</td>
</tr>
<tr>
<td>unfolding refl-on-def</td>
</tr>
<tr>
<td>by simp</td>
</tr>
<tr>
<td>ultimately show $\exists Q'. \ P \mapsto Cal* \ Q' \land P'' \mapsto&lt;Rel,Cal&gt; \ Q'$</td>
</tr>
<tr>
<td>by blast</td>
</tr>
</tbody>
</table>

next

<table>
<thead>
<tr>
<th>case (trans $P Q R$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>assume $P \mapsto Cal* \ P'$ and $\bigwedge P'. \ P \mapsto Cal* \ P' \implies \exists Q'. \ Q \mapsto Cal* \ Q' \land P' \mapsto&lt;Rel,Cal&gt; \ Q'$</td>
</tr>
<tr>
<td>from this obtain $Q'$ where $A1: Q \mapsto Cal* \ Q'$ and $A2: P' \mapsto&lt;Rel,Cal&gt; \ Q'$</td>
</tr>
<tr>
<td>by blast</td>
</tr>
<tr>
<td>assume $\bigwedge Q'. \ Q \mapsto Cal* \ Q' \implies \exists R'. \ R \mapsto Cal* \ R' \land Q' \mapsto&lt;Rel,Cal&gt; \ R'$</td>
</tr>
<tr>
<td>with $A1$ obtain $R'$ where $A3: R \mapsto Cal* \ R'$ and $A4: Q' \mapsto&lt;Rel,Cal&gt; \ R'$</td>
</tr>
<tr>
<td>by blast</td>
</tr>
<tr>
<td>from $A2 A4$ have $P' \mapsto&lt;Rel,Cal&gt; \ R'$</td>
</tr>
<tr>
<td>by (rule stepsClosure.trans)</td>
</tr>
<tr>
<td>with $A3$ show $\exists R'. \ R \mapsto Cal* \ R' \land P' \mapsto&lt;Rel,Cal&gt; \ R'$</td>
</tr>
<tr>
<td>by blast</td>
</tr>
<tr>
<td>qed</td>
</tr>
<tr>
<td>qed</td>
</tr>
</tbody>
</table>

If $Rel$ is a weak simulation and its inverse is a weak contrasimulation, then the steps closure of $Rel$ is a contrasimulation.

lemma inverse-contrasimulation-impl-reverse-pair-in-steps-closure:

<table>
<thead>
<tr>
<th>fixes $Rel :: ('a \times 'a) set$</th>
</tr>
</thead>
<tbody>
<tr>
<td>and $Cal :: 'a processCalculus$</td>
</tr>
<tr>
<td>and $P Q :: 'a$</td>
</tr>
<tr>
<td>assumes $con :: weak-reduction-contrasimulation (Rel^{-1}) Cal$</td>
</tr>
<tr>
<td>and $pair :: (P, Q) \in Rel$</td>
</tr>
<tr>
<td>shows $Q \mapsto&lt;Rel,Cal&gt; P$</td>
</tr>
</tbody>
</table>

proof —

| from $pair$ have $(Q, P) \in Rel^{-1}$ |
| by simp |
| moreover have $Q \mapsto Cal* \ Q$ |
| by (rule steps-refl) |
| ultimately obtain $P'$ where $A1: P \mapsto Cal* P'$ and $A2: (P', Q) \in Rel^{-1}$ |
| using $con$ |
| by blast |
| from $A2$ have $Q \mapsto<Rel,Cal> P'$ |
| by (simp add: stepsClosure.rel) |

42
moreover from A1 have $P' \mathcal{R} \rightarrow<\mathsf{Rel}, \mathsf{Cal}> P$
by (rule stepsClosure.steps)
ultimately show $Q \mathcal{R} \rightarrow<\mathsf{Rel}, \mathsf{Cal}> P$
by (rule stepsClosure.trans)
qed

lemma simulation-and-inverse-contrasimulation-impl-steps-closure-is-contrasimulation:
fixes $\mathsf{Rel} :: (a \times a)$ set
and $\mathsf{Cal} :: \text{'a processCalculus}$
assumes sim: weak-reduction-simulation $\mathsf{Rel} \mathsf{Cal}$
and con: weak-reduction-contrasimulation ($\mathsf{Rel}^{-1}$) $\mathsf{Cal}$
shows weak-reduction-contrasimulation ($\mathsf{stepsClosure} \mathsf{Rel} \mathsf{Cal}$) $\mathsf{Cal}$
proof clarify
fix $P$ $Q$ $P'$
assume $P \mathcal{R} \rightarrow<\mathsf{Rel}, \mathsf{Cal}> Q$ and $P \rightarrow<\mathsf{Cal}> P'$$thus \exists Q'. Q \rightarrow<\mathsf{Cal}> Q' \land Q' \mathcal{R} \rightarrow<\mathsf{Rel}, \mathsf{Cal}> P'$
proof (induct arbitrary: $P'$)
case (rel $P$ $Q$)
assume $(P, Q) \in \mathsf{Rel}$ and $P \rightarrow<\mathsf{Cal}> P'$$with sim obtain $Q'$ where $A1� Q \rightarrow<\mathsf{Cal}> Q'$ and $A2¶ (P', Q') \in \mathsf{Rel}$
by blast
from $A2$ con have $Q' \mathcal{R} \rightarrow<\mathsf{Rel}, \mathsf{Cal}> P'$
using inverse-contrasimulation-impl-reverse-pair-in-steps-closure[where $\mathsf{Rel} = \mathsf{Rel}$]
by blast
with $A1$ show $\exists Q'. Q \rightarrow<\mathsf{Cal}> Q' \land Q' \mathcal{R} \rightarrow<\mathsf{Rel}, \mathsf{Cal}> P'$
by blast
next
case (steps $P$ $P'$ $P''$)
assume $P \rightarrow<\mathsf{Cal}> P'$ and $P' \rightarrow<\mathsf{Cal}> P''$ hence $P \rightarrow<\mathsf{Cal}> P''$
by (rule steps-add)
thus $\exists Q'. P \rightarrow<\mathsf{Cal}> Q' \land Q' \mathcal{R} \rightarrow<\mathsf{Rel}, \mathsf{Cal}> P''$
using stepsClosure-refl[where $\mathsf{Rel} = \mathsf{Rel}$ and $\mathsf{Cal} = \mathsf{Cal}$]
unfolding refl-on-def
by blast
next
case (trans $P$ $Q$ $R$)
assume $(P', P) \rightarrow<\mathsf{Cal}> Q' \Rightarrow \exists Q'. Q \rightarrow<\mathsf{Cal}> Q' \land Q' \mathcal{R} \rightarrow<\mathsf{Rel}, \mathsf{Cal}> P'$
and $P \rightarrow<\mathsf{Cal}> P'$
from this obtain $Q'$ where $A1¶ Q \rightarrow<\mathsf{Cal}> Q'$ and $A2¶ Q' \mathcal{R} \rightarrow<\mathsf{Rel}, \mathsf{Cal}> P'$
by blast
assume $(Q', Q) \rightarrow<\mathsf{Cal}> Q' \Rightarrow \exists R'. R \rightarrow<\mathsf{Cal}> R' \land R' \mathcal{R} \rightarrow<\mathsf{Rel}, \mathsf{Cal}> Q'$
with $A1$ obtain $R'$ where $A3¶ R \rightarrow<\mathsf{Cal}> R'$ and $A4¶ R' \mathcal{R} \rightarrow<\mathsf{Rel}, \mathsf{Cal}> Q'$
by blast
from $A4$ $A2$ have $R' \mathcal{R} \rightarrow<\mathsf{Rel}, \mathsf{Cal}> P'$
by (rule stepsClosure.trans)
with $A3$ show $\exists R'. R \rightarrow<\mathsf{Cal}> R' \land R' \mathcal{R} \rightarrow<\mathsf{Rel}, \mathsf{Cal}> P'$
by blast
qed

Accordingly, if Rel is a weak simulation and its inverse is a weak contrasimulation, then the steps closure of Rel is a coupled simulation.

lemma simulation-and-inverse-contrasimulation-impl-steps-closure-is-coupled-simulation:
fixes $\mathsf{Rel} :: (a \times a)$ set
and $\mathsf{Cal} :: \text{'a processCalculus}$
assumes sim: weak-reduction-simulation $\mathsf{Rel} \mathsf{Cal}$
and con: weak-reduction-contrasimulation ($\mathsf{Rel}^{-1}$) $\mathsf{Cal}$
shows weak-reduction-coupled-simulation ($\mathsf{stepsClosure} \mathsf{Rel} \mathsf{Cal}$) $\mathsf{Cal}$
using sim con simulation-and-inverse-contrasimulation-impl-steps-closure-is-contrasimulation
lemma stepsClosure-simulation-impl-refl-trans-closure-of-Rel:

\begin{proof}
\begin{itemize}
  \item \hspace{1cm} Assume \((P', Q') \in \text{Rel}\)
  \item \hspace{1cm} With \(A2\) have \(P' \mapsto \text{Cal} \ast P'' \mapsto (\exists Q'. \ Q \mapsto \text{Cal} \ast Q' \land (P', Q') \in \text{Rel}^*)\)
  \item \hspace{1cm} By \text{blast}
  \item \hspace{1cm} Thus \(\exists P''. P \mapsto \text{Cal} \ast P'' \mapsto (\exists Q'. Q \mapsto \text{Cal} \ast Q' \land (P', Q') \in \text{Rel}^*)\)
  \item \hspace{1cm} By \text{blast}
\end{itemize}
\end{proof}

qed

next

\begin{proof}
\begin{itemize}
  \item \hspace{1cm} Assume \(A: P \mapsto \text{Cal} \ast P'\)
  \item \hspace{1cm} Show \(\forall P'''. P' \mapsto \text{Cal} \ast P''' \mapsto (\exists Q'. Q \mapsto \text{Cal} \ast Q' \land (P'', Q') \in \text{Rel}^*)\)
  \item \hspace{1cm} By \text{clarify}
  \item \hspace{1cm} Fix \(P''\)
  \item \hspace{1cm} With \(A\) have \(P' \mapsto \text{Cal} \ast P''\)
  \item \hspace{1cm} By \text{rule steps-add}
  \item \hspace{1cm} Moreover have \((P'', P') \in \text{Rel}^*)\)
  \item \hspace{1cm} By \text{simp}
  \item \hspace{1cm} Ultimately show \(\exists Q'. P \mapsto \text{Cal} \ast Q' \land (P'', Q') \in \text{Rel}^*)\)
  \item \hspace{1cm} By \text{blast}
\end{itemize}
\end{proof}

qed

next

\begin{proof}
\begin{itemize}
  \item \hspace{1cm} Assume \(A1: \forall P'. P \mapsto \text{Cal} \ast P' \mapsto (\exists Q'. Q \mapsto \text{Cal} \ast Q' \land (P', Q') \in \text{Rel}^*)\)
  \item \hspace{1cm} And \(A2: \forall Q'. Q \mapsto \text{Cal} \ast Q' \mapsto (\exists R'. R \mapsto \text{Cal} \ast R' \land (Q', R') \in \text{Rel}^*)\)
  \item \hspace{1cm} Show \(\forall P'. P \mapsto \text{Cal} \ast P' \mapsto (\exists R'. R \mapsto \text{Cal} \ast R' \land (P', R') \in \text{Rel}^*)\)
  \item \hspace{1cm} By \text{clarify}
  \item \hspace{1cm} Fix \(P''\)
  \item \hspace{1cm} With \(A1\) obtain \(Q'\) where \(A3: Q \mapsto \text{Cal} \ast Q'\) and \(A4: (P', Q') \in \text{Rel}^*\)
  \item \hspace{1cm} By \text{blast}
  \item \hspace{1cm} From \(A2 \ A3\) obtain \(R'\) where \(A5: R \mapsto \text{Cal} \ast R'\) and \(A6: (Q', R') \in \text{Rel}^*\)
  \item \hspace{1cm} By \text{blast}
  \item \hspace{1cm} From \(A4 \ A6\) have \((P', R') \in \text{Rel}^*)\)
  \item \hspace{1cm} By \text{simp}
  \item \hspace{1cm} With \(A5\) show \(\exists R'. R \mapsto \text{Cal} \ast R' \land (P', R') \in \text{Rel}^*)\)
  \item \hspace{1cm} By \text{blast}
\end{itemize}
\end{proof}

qed

moreover have \(P \mapsto \text{Cal} \ast P\)
\hspace{1cm} By \text{rule steps-refl}

ultimately show \(?\) thesis
\hspace{1cm} By \text{blast}

qed
lemma stepsClosure-contrasimulation-impl-refl-trans-closure-of-Rel:
  fixes Rel :: ('a × 'a) set
  and Cal :: 'a processCalculus
  and P Q :: 'a
  assumes A1: P Rel< Cal> Q
  and A2: weak-reduction-contrasimulation Rel Cal
  shows ∃ Q'. Q Cal→ Cal* Q' ∧ (Q', P) ∈ Rel*
proof
  have ∀ P'. P Cal→ Cal* P' Cal→ (Q Q' Cal→ Q Cal* Cal Q' ∧ (Q', P) ∈ Rel*)
  using A1
proof induction
  show ∀ P. P Cal→ Cal* P Cal→ (∃ Q'. P Cal→ Q Cal* Q' ∧ (Q', P) ∈ Rel*)
proof clarify
  fix P'
  assume P' Cal→ Cal* P'
  with A1 obtain Q' where A3: Q Cal→ Cal* Q' and A4: (Q', P) ∈ Rel*
  by blast
  from A2 A3 obtain R' where A5: R Cal→ Cal* R' and A6: (R', Q') ∈ Rel*
  by blast
  from A4 A6 have (R', P') ∈ Rel*
  by simp
  with A5 show ∃ R'. R Cal→ Cal* R' ∧ (R', P') ∈ Rel*
  by blast
  qed
  qed
next
  case (trans P Q R)
  assume A1: ∀ P'. P Cal→ Cal* P' Cal→ (Q Q' Cal→ Q Cal* Cal Q' ∧ (Q', P) ∈ Rel*)
  and A2: ∀ Q'. Q Cal→ Cal* Q' Cal→ (Q Q' Cal→ Q Cal* Cal Q' ∧ (Q', P) ∈ Rel*)
  show ∀ P'. P Cal→ Cal* P' Cal→ (Q Q' Cal→ Q Cal* Cal Q' ∧ (Q', P) ∈ Rel*)
proof clarify
  fix P
  assume P Cal→ Cal* P
  with A1 obtain Q' where A3: Q Cal→ Cal* Q' and A4: (Q', P) ∈ Rel*
  by blast
  from A2 A3 obtain R' where A5: R Cal→ Cal* R' and A6: (R', Q') ∈ Rel*
  by blast
  from A4 A6 have (R', P') ∈ Rel*
  by simp
  with A5 show ∃ R'. R Cal→ Cal* R' ∧ (R', P') ∈ Rel*
  by blast
  qed
  qed
moreover have P Cal→ Cal* P
  by (rule steps-refl)
ultimately show ?thesis
  by blast
  qed
qed

lemma stepsClosure-contrasimulation-of-inverse-impl-refl-trans-closure-of-Rel:
  fixes Rel :: ('a × 'a) set
  and Cal :: 'a processCalculus
  and P Q :: 'a
  assumes A1: P Rel< Cal> Q
  and A2: weak-reduction-contrasimulation (Rel⁻¹) Cal

shows \( \exists Q'. Q \rightarrow Cal* \ Q' \land (P', Q') \in Rel^* \)

proof -

have \( \forall P'. P \rightarrow Cal* \ P' \rightarrow (\exists Q'. Q \rightarrow Cal* \ Q' \land (P', Q') \in Rel^*) \)

using A1

proof induct

case (rel P Q)

assume \((P, Q) \in Rel^{-1}\)

with A2 have \( \forall P'. P \rightarrow Cal* \ P' \rightarrow (\exists Q'. Q \rightarrow Cal* \ Q' \land (Q', P') \in Rel^{-1}) \)

by blast

thus \( \forall P'. P \rightarrow Cal* \ P' \rightarrow (\exists Q'. Q \rightarrow Cal* \ Q' \land (P', Q') \in Rel^*) \)

by blast

next

case (steps P P')

assume \( A: P \rightarrow Cal* \ P' \)

show \( \forall P''. P' \rightarrow Cal* \ P'' \rightarrow (\exists Q'. P \rightarrow Cal* \ Q' \land (P'', Q') \in Rel^*) \)

proof clarify

fix \( P'' \)

assume \( P' \rightarrow Cal* \ P'' \)

with A have \( P \rightarrow Cal* \ P'' \)

by (rule steps-add)

moreover have \( (P'', Q') \in Rel^* \)

by simp

ultimately show \( \exists Q'. P \rightarrow Cal* \ Q' \land (P'', Q') \in Rel^* \)

by blast

qed

next

case (trans P Q R)

assume \( A1: \forall P'. P \rightarrow Cal* \ P' \rightarrow (\exists Q'. Q \rightarrow Cal* \ Q' \land (P', Q') \in Rel^*) \)

and \( A2: \forall Q'. Q \rightarrow Cal* \ Q' \rightarrow (\exists R'. R \rightarrow Cal* \ R' \land (Q', R') \in Rel^*) \)

show \( \forall P'. P \rightarrow Cal* \ P' \rightarrow (\exists R'. R \rightarrow Cal* \ R' \land (P', R') \in Rel^*) \)

proof clarify

fix \( P' \)

with A1 obtain \( Q' \) where \( A3: Q \rightarrow Cal* \ Q' \) and \( A4: (P', Q') \in Rel^* \)

by blast

from A3 A2 obtain \( R' \) where \( A5: R \rightarrow Cal* \ R' \) and \( A6: (Q', R') \in Rel^* \)

by blast

from A4 A6 have \( (P', R') \in Rel^* \)

by simp

with A5 show \( \exists R'. R \rightarrow Cal* \ R' \land (P', R') \in Rel^* \)

by blast

qed

qed

moreover have \( P \rightarrow Cal* \ P \)

by (rule steps-refl)

ultimately show \( \exists \)

thesis

by blast

qed

end

theory Encodings

imports ProcessCalculi

begin

4 Encodings

In the simplest case an encoding from a source into a target language is a mapping from source into target terms. Encodability criteria describe properties on such mappings. To analyse encodability criteria we map them on conditions on relations between source and target terms. More precisely, we
consider relations on pairs of the disjoint union of source and target terms. We denote this disjoint union of source and target terms by Proc.

**datatype** ('procS, 'procT) Proc =

  SourceTerm 'procS |
  TargetTerm 'procT

**definition** STCal

  :: 'procS processCalculus ⇒ 'procT processCalculus
     ⇒ (('procS, 'procT) Proc) processCalculus

  where

  STCal Source Target =
     \( \langle \text{Reductions} = \lambda P. P', \langle \exists SP SP', P = \text{SourceTerm} SP \wedge P' = \text{SourceTerm} SP' \wedge \text{Reductions} \text{Source} SP SP' \rangle \rangle \vee \langle \exists TP TP', P = \text{TargetTerm} TP \wedge P' = \text{TargetTerm} TP' \wedge \text{Reductions} \text{Target} TP TP' \rangle \rangle \)

**definition** STCalWB

  :: ('procS, 'barbs) calculusWithBarbs ⇒ ('procT, 'barbs) calculusWithBarbs
     ⇒ (('procS, 'procT) Proc, 'barbs) calculusWithBarbs

  where

  STCalWB Source Target =
     \( \langle \text{Calculus} = \text{STCal} \text{calculusWithBarbs. Calculus Source} \rangle \langle \text{calculusWithBarbs. Calculus Target} \rangle, \text{HasBarb} = \lambda a. \langle \exists SP SP', P = \text{SourceTerm} SP \wedge (\text{calculusWithBarbs. HasBarb Source}) SP a \rangle \rangle \vee \langle \exists TP TP', P = \text{TargetTerm} TP \wedge (\text{calculusWithBarbs. HasBarb Target}) TP a \rangle \rangle \)

An encoding consists of a source language, a target language, and a mapping from source into target terms.

**locale** encoding =

  fixes Source :: 'procS processCalculus
  and Target :: 'procT processCalculus
  and Enc :: 'procS ⇒ 'procT
begin

abbreviation enc :: 'procS ⇒ 'procT ([1] [65] 70) where
  \([S]\) ≡ Enc S

abbreviation issSource :: ('procS, 'procT) Proc ⇒ bool (- ∈ ProcS [70] 80) where
  P ∈ ProcS ≡ (\exists S. P = \text{SourceTerm} S)

abbreviation issTarget :: ('procS, 'procT) Proc ⇒ bool (- ∈ ProcT [70] 80) where
  P ∈ ProcT ≡ (\exists T. P = \text{TargetTerm} T)

abbreviation getSource

  :: 'procS ⇒ ('procS, 'procT) Proc ⇒ bool (- ∈ S - [70] 80)

  where

  S ∈ S P ≡ (P = \text{SourceTerm} S)

abbreviation getTarget

  :: 'procT ⇒ ('procS, 'procT) Proc ⇒ bool (- ∈ T - [70] 80)

  where

  T ∈ T P ≡ (P = \text{TargetTerm} T)
A step of a term in Proc is either a source term step or a target term step.

**abbreviation** stepST

  :: ('procS, 'procT) Proc ⇒ ('procS, 'procT) Proc ⇒ bool (- → ST - [70] 80)

  where

  P → ST P' ≡ \( \langle \exists S S', S ∈ S P ∧ S' ∈ S P' ∧ S → \text{Source} S' \rangle \rangle \vee \langle \exists T T'. T ∈ T P ∧ T' ∈ T P' ∧ T → \text{Target} T' \rangle \rangle \)

**lemma** stepST-STCal-step:

  fixes P P' :: ('procS, 'procT) Proc
shows $P \rightsquigarrow (ST\text{Cal} \text{ Source Target})$ $P' = P \rightsquigarrow ST \ P'$
by (simp add: STCal-def)

**lemma STStep-step:**

fixes $S :: 'procS$
and $T :: 'procT$
and $P' :: ('procS, 'procT) \ Proc$

shows $SourceTerm S \rightsquigarrow ST \ P' = (\exists S'. \ S' \in S \ P' \wedge S \rightsquigarrow Source \ S')$
and $TargetTerm T \rightsquigarrow ST \ P' = (\exists T'. \ T' \in T \ P' \wedge T \rightarrow Target \ T')$
by blast+

**lemma STCal-step:**

fixes $S :: 'procS$
and $T :: 'procT$
and $P' :: ('procS, 'procT) \ Proc$

shows $SourceTerm S \rightarrow (ST\text{Cal} \text{ Source Target}) \ P' = (\exists S'. \ S' \in S \ P' \wedge S \rightarrow Source \ S')$
and $TargetTerm T \rightarrow (ST\text{Cal} \text{ Source Target}) \ P' = (\exists T'. \ T' \in T \ P' \wedge T \rightarrow Target \ T')$
by (simp add: STCal-def)+

A sequence of steps of a term in Proc is either a sequence of source term steps or a sequence of target term steps.

**abbreviation stepsST**

$:: ('procS, 'procT) \ Proc \Rightarrow ('procS, 'procT) \ Proc \Rightarrow bool (\cdot \rightarrow ST* - [70, 70] 80)$

where
$P \rightarrow ST* \ P' \equiv$
$(\exists S S'. \ S \in S \ P \wedge S' \in S \ P' \wedge S \rightarrow Source \ S') \lor (\exists T T'. \ T \in T \ P \wedge T' \in T \ P' \wedge T \rightarrow Target \ T')$

**lemma STSteps-steps:**

fixes $S :: 'procS$
and $T :: 'procT$
and $P' :: ('procS, 'procT) \ Proc$

shows $SourceTerm S \rightarrow (ST\text{Steps} \text{ Source Target}) \ P' = (\exists S'. \ S' \in S \ P' \wedge S \rightarrow Source* \ S')$
and $TargetTerm T \rightarrow (ST\text{Steps} \text{ Source Target}) \ P' = (\exists T'. \ T' \in T \ P' \wedge T \rightarrow Target* \ T')$
by blast+

**lemma STCal-steps:**

fixes $S :: 'procS$
and $T :: 'procT$
and $P' :: ('procS, 'procT) \ Proc$

shows $SourceTerm S \rightarrow (ST\text{Cal} \text{ Source Target})* \ P' = (\exists S'. \ S' \in S \ P' \wedge S \rightarrow Source* \ S')$
and $TargetTerm T \rightarrow (ST\text{Cal} \text{ Source Target})* \ P' = (\exists T'. \ T' \in T \ P' \wedge T \rightarrow Target* \ T')$

**proof auto**

assume $SourceTerm S \rightarrow (ST\text{Cal} \text{ Source Target})* \ P''$
from this obtain $n$ where $SourceTerm S \rightarrow (ST\text{Cal} \text{ Source Target})^n \ P'$
by (auto simp add: steps-def)
thus $\exists S'. \ S' \in S \ P' \wedge S \rightarrow Source* \ S'$
**proof (induct $n$ arbitrary: $P'$)**

**case 0**

assume $SourceTerm S \rightarrow (ST\text{Cal} \text{ Source Target})^0 \ P''$

**hence** $S \in S \ P'$$
by simp

**moreover have** $S \rightarrow Source* \ S$

by (rule steps-refl)

**ultimately show** $\exists S'. \ S' \in S \ P' \wedge S \rightarrow Source* \ S'$

by blast

**next**

**case (Suc $n$ $P''')**

assume $SourceTerm S \rightarrow (ST\text{Cal} \text{ Source Target})^{Suc \ n} \ P''''$
from this obtain $P'$ where $A1: SourceTerm S \rightarrow (ST\text{Cal} \text{ Source Target})^n \ P'$$and $A2: P', \rightarrow (ST\text{Cal} \text{ Source Target}) \ P''$$
by auto
assume \( P' \). SourceTerm S \( \mapsto (STCal Source Target)^n \) P' \( \Rightarrow \exists S'. S' \in S P' \land S \mapsto Source* S' \)
with A1 obtain S' where A3: S' \( \in S \) P' and A4: S \( \mapsto Source* S' \)
by blast
from A2 A3 obtain S'' where A5: S'' \( \in S \) P'' and A6: S' \( \mapsto Source S'' \)
using STCal-step(1)[where S=S' and P=P'']
by blast
from A4 A6 have S \( \mapsto Source* S'' \)
using step-to-steps[where Cal=Source and P=S' and P'=S'']
by (simp add: steps-add[where Cal=Source and P=S and Q=S' and R=S''])
with A5 show \( \exists S''. S'' \in S P'' \land S \mapsto Source* S'' \)
by blast
qed
next
fix S'
assume S \( \mapsto Source* S' \)
from this obtain n where S \( \mapsto Source^n S' \)
by (auto simp add: steps-def)
thus SourceTerm S \( \mapsto (STCal Source Target)^* \) (SourceTerm S')
proof (induct n arbitrary: S')
case 0
assume S \( \mapsto Source^0 S' \)
hence S = S'
by auto
thus SourceTerm S \( \mapsto (STCal Source Target)^* \) (SourceTerm S')
by (simp add: steps-refl)
next
case (Suc n S'')
assume S \( \mapsto Source^{Suc n} S'' \)
from this obtain S' where B1: S \( \mapsto Source^n S' \) and B2: S' \( \mapsto Source S'' \)
by auto
assume \( \forall S'. S \mapsto Source^n S' \Rightarrow SourceTerm S \mapsto (STCal Source Target)^* \) (SourceTerm S')
with B1 have SourceTerm S \( \mapsto (STCal Source Target)^* \) (SourceTerm S')
by blast
moreover from B2 have SourceTerm S' \( \mapsto (STCal Source Target)^* \) (SourceTerm S'')
using step-to-steps[where Cal=STCal Source Target and P=SourceTerm S']
by (simp add: STCal-def)
ultimately show SourceTerm S \( \mapsto (STCal Source Target)^* \) (SourceTerm S'')
by (rule steps-add)
qed
next
assume TargetTerm T \( \mapsto (STCal Source Target)^* P' \)
from this obtain n where TargetTerm T \( \mapsto (STCal Source Target)^n P' \)
by (auto simp add: steps-def)
thus \( \exists T'. T' \in T P' \land T \mapsto Target* T' \)
proof (induct n arbitrary: P')
case 0
assume TargetTerm T \( \mapsto (STCal Source Target)^0 P' \)
hence T \( \in T P' \)
by simp
moreover have T \( \mapsto Target* T \)
by (rule steps-refl)
ultimately show \( \exists T'. T' \in T P' \land T \mapsto Target* T' \)
by blast
next
case (Suc n P'')
assume TargetTerm T \( \mapsto (STCal Source Target)^{Suc n} P'' \)
from this obtain P' where A1: TargetTerm T \( \mapsto (STCal Source Target)^n P' \)
and A2: P' \( \mapsto (STCal Source Target) P'' \)
by auto
assume $\land P', \text{TargetTerm } T \mapsto (\text{STCal Source Target})^n P' \Rightarrow \exists T'. T' \in T P' \land T \mapsto \text{Target* } T'$

with A1 obtain $T'$ where A3: $T' \in T P'$ and A4: $T \mapsto \text{Target* } T'$

by blast

from A2 A3 obtain $T''$ where A5: $T'' \in T P''$ and A6: $T' \mapsto \text{Target* } T''$

using STCal-step(2)[where $T=T'$ and $P'=P''$]

by blast

from A4 A6 have $T \mapsto \text{Target* } T''$

using step-to-steps[where $\text{Cal}=\text{Target and } P=T$ and $P'=T''$]

by (simp add: steps-add[where $\text{Cal}=\text{Target and } P=T$ and $Q=T'$ and $R=T''$])

with A5 show $\exists T''$. $T'' \in T P'' \land T \mapsto \text{Target* } T''$

by blast

qed

next

fix $T'$

assume $T \mapsto \text{Target* } T'$

from this obtain $n$ where $T \mapsto \text{Target}^n T'$

by (auto simp add: steps-def)

thus TargetTerm $T \mapsto (\text{STCal Source Target})* (\text{TargetTerm } T')$

proof (induct $n$ arbitrary: $T'$)

case 0

assume $T \mapsto \text{Target}^0 T'$

hence $T = T'$

by auto

thus TargetTerm $T \mapsto (\text{STCal Source Target})* (\text{TargetTerm } T')$

by (simp add: steps-refl)

next

case (Suc $n$ $T''$)

assume $T \mapsto \text{Target}^{\text{Suc } n} T''$

from this obtain $T'$ where $B1$: $T \mapsto \text{Target}^n T'$ and $B2$: $T' \mapsto \text{Target} T''$

by auto

assume $\land T'. T \mapsto \text{Target}^n T' \Rightarrow \text{TargetTerm } T \mapsto (\text{STCal Source Target})* (\text{TargetTerm } T')$

with $B1$ have TargetTerm $T \mapsto (\text{STCal Source Target})* (\text{TargetTerm } T')$

by blast

moreover from $B2$ have TargetTerm $T' \mapsto (\text{STCal Source Target})* (\text{TargetTerm } T'')$

using step-to-steps[where $\text{Cal}=\text{STCal Source Target}$ and $P=\text{TargetTerm } T'$]

by (simp add: STCal-def)

ultimately show TargetTerm $T \mapsto (\text{STCal Source Target})* (\text{TargetTerm } T'')$

by (rule steps-add)

qed

qed

lemma stepsST-STCal-steps:

fixes $P P'$ :: ("procS", "procT") Proc

shows $P \mapsto (\text{STCal Source Target})* P' = P \mapsto \text{ST}* P'$

proof (cases $P$

case (SourceTerm $SP$)

assume $SP \in S P$

thus $P \mapsto (\text{STCal Source Target})* P' = P \mapsto \text{ST}* P'$

using STCal-steps(1)[where $S=SP$ and $P'=P'$] STSteps-steps(1)[where $S=SP$ and $P'=P'$]

by blast

next

case (TargetTerm $TP$)

assume $TP \in T P$

thus $P \mapsto (\text{STCal Source Target})* P' = P \mapsto \text{ST}* P'$

using STCal-steps(2)[where $T=TP$ and $P'=P'$] STSteps-steps(2)[where $T=TP$ and $P'=P'$]

by blast

qed

lemma stepsST-refl:

fixes $P$ :: ("procS", "procT") Proc
shows \( P \mapsto ST^* P \)
by (cases \( P \), simp-all add: steps-refl)

**lemma stepsST-add:**
\[
\begin{align*}
\text{fixes } P \text{ Q R} &:: ('procS, 'procT) Proc \\
\text{assumes A1: } P &\mapsto ST^* Q \\
\text{and A2: } Q &\mapsto ST^* R \\
\text{shows } P &\mapsto ST^* R \\
\text{proof } - \\
\text{from A1 have } P &\mapsto (STCal Source Target)^* Q \\
\text{by (simp add: stepsST-STCal-steps)} \\
\text{moreover from A2 have } Q &\mapsto (STCal Source Target)^* R \\
\text{by (simp add: stepsST-STCal-steps)} \\
\text{ultimately have } P &\mapsto (STCal Source Target)^* R \\
\text{by (rule steps-add)} \\
\text{thus } P &\mapsto ST^* R \\
\text{by (simp add: stepsST-STCal-steps)} \\
\text{qed}
\end{align*}
\]

A divergent term of Proc is either a divergent source term or a divergent target term.

**abbreviation divergentST**
\[
:: ('procS, 'procT) Proc \Rightarrow bool (\cdot \mapsto ST\omega [70] 80)
\]
where
\[
P \mapsto ST\omega \equiv (\exists S. S \in S P \land S \mapsto (Source)^\omega) \lor (\exists T. T \in T P \land T \mapsto (Target)^\omega)
\]

**lemma STCal-divergent:**
\[
\begin{align*}
\text{fixes } S &:: 'procS \\
\text{and } T &:: 'procT \\
\text{shows } SourceTerm S &\mapsto (STCal Source Target)^\omega = S \mapsto (Source)^\omega \\
\text{and } TargetTerm T &\mapsto (STCal Source Target)^\omega = T \mapsto (Target)^\omega \\
\text{using } STCal-steps \\
\text{by (auto simp add: STCal-def divergent-def)}
\end{align*}
\]

**lemma divergentST-STCal-divergent:**
\[
\begin{align*}
\text{fixes } P &:: ('procS, 'procT) Proc \\
\text{shows } P &\mapsto (STCal Source Target)^\omega = P \mapsto ST\omega \\
\text{proof (cases } P) \\
\text{case } (SourceTerm SP) \\
\text{assume } SP \in S P \\
\text{thus } P &\mapsto (STCal Source Target)^\omega = P \mapsto ST\omega \\
\text{using } STCal-divergent(1) \\
\text{by simp} \\
\text{next } \\
\text{case } (TargetTerm TP) \\
\text{assume } TP \in T P \\
\text{thus } P &\mapsto (STCal Source Target)^\omega = P \mapsto ST\omega \\
\text{using } STCal-divergent(2) \\
\text{by simp} \\
\text{qed}
\end{align*}
\]

Similar to relations we define what it means for an encoding to preserve, reflect, or respect a predicate. An encoding preserves some predicate \( P \) if \( P(S) \) implies \( P(\text{enc } S) \) for all source terms \( S \).

**abbreviation enc-preserves-pred**
\[
:: ('procS, 'procT) Proc \Rightarrow bool (\cdot \Rightarrow \text{bool}) \Rightarrow \text{bool where}
enc-preserves-pred Pred \equiv \forall S. Pred (SourceTerm S) \mapsto Pred (TargetTerm ([S]))
\]

**abbreviation enc-preserves-binary-pred**
\[
:: ('procS, 'procT) Proc \Rightarrow 'b \Rightarrow bool \\
\text{where}
enc-preserves-binary-pred Pred \equiv \forall S x. Pred (SourceTerm S) x \mapsto Pred (TargetTerm ([S])) x
\]

An encoding reflects some predicate \( P \) if \( P(S) \) implies \( P(\text{enc } S) \) for all source terms \( S \).
abbreviation enc-reflects-pred :: (\('procS, 'procT\) Proc ⇒ bool) ⇒ bool where
  enc-reflects-pred Pred ≡ ∀ S. Pred (TargetTerm ([S])) → Pred (SourceTerm S)

abbreviation enc-reflects-binary-pred
  :: (('procS, 'procT) Proc ⇒ 'b ⇒ bool) ⇒ bool where
  enc-reflects-binary-pred Pred ≡ ∀ S x. Pred (TargetTerm ([S]) x) → Pred (SourceTerm S) x

An encoding respects a predicate if it preserves and reflects it.

abbreviation enc-respects-pred :: (('procS, 'procT) Proc ⇒ bool) ⇒ bool where
  enc-respects-pred Pred ≡ enc-preserves-pred Pred ∧ enc-reflects-pred Pred

abbreviation enc-respects-binary-pred
  :: (('procS, 'procT) Proc ⇒ 'b ⇒ bool) ⇒ bool where
  enc-respects-binary-pred Pred ≡ enc-preserves-binary-pred Pred ∧ enc-reflects-binary-pred Pred

end

To compare source terms and target terms w.r.t. their barbs or observables we assume that each
languages defines its own predicate for the existence of barbs.

locale encoding-wrt-barbs =
  encoding Source Target Enc
  for Source :: 'procS processCalculus
  and Target :: 'procT processCalculus
  and Enc :: 'procS ⇒ 'procT +
  fixes SWB :: ('procS, 'barbs) calculusWithBarbs
  and TWB :: ('procT, 'barbs) calculusWithBarbs
  assumes calS: calculusWithBarbs.Calculus SWB = Source
  and calT: calculusWithBarbs.Calculus TWB = Target
begin

lemma STCalWB-STCal:
  shows Calculus (STCalWB SWB TWB) = STCal Source Target
  unfolding STCalWB-def using calS calT
  by auto

We say a term P of Proc has some barbs a if either P is a source term that has barb a or P is a target
term that has the barb b. For simplicity we assume that the sets of barbs is large enough to contain
all barbs of the source terms, the target terms, and all barbs they might have in common.

abbreviation hasBarbST
  :: (\('procS, 'procT\) Proc ⇒ 'barbs ⇒ bool) where
  P ↓. a ≡ (∃ S. S ∈ S P ∧ S ↓<SWB>a) ∨ (∃ T. T ∈ T P ∧ T ↓<TWB>a)

lemma STCalWB-hasBarbST:
  fixes P :: ('procS, 'procT) Proc
  and a :: 'barbs
  shows P ↓<STCalWB SWB TWB>a = P ↓. a
  by (simp add: STCalWB-def)

lemma preservation-of-barbs-in-barbed-encoding:
  fixes Rel :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) set
  and P Q :: ('procS, 'procT) Proc
  and a :: 'barbs
  assumes preservation: rel-preserves-barbs Rel (STCalWB SWB TWB)
  and rel: (P, Q) ∈ Rel
  and barb: P ↓. a
  shows Q ↓. a

52
proof
\[\text{− weak-preservation-of-barbs-in-barbed-encoding lemma}\]

\[\text{proof}\]
\[\text{STCalWB-reachesBarbST}\]
\[\text{abbreviation}\]
\[\text{reachesBarbST}\]
that reaches a.
A term P of Proc reaches a barb a if either P is a source term that reaches a or P is a target term that reaches a.

\[\text{lemma reflection-of-barbs-in-barbed-encoding}\]
\[\text{fixes Rel :: } ((\text{procS}, \text{procT}) \text{ Proc} \times (\text{procS}, \text{procT}) \text{ Proc}) \text{ set}\]
\[\text{and P Q :: (procS, procT) Proc}\]
\[\text{and a :: 'barbs}\]
\[\text{assumes reflection: rel-reflects-barbs Rel (STCalWB SWB TWB)}\]
\[\text{and rel: } (P, Q) \in \text{Rel}\]
\[\text{and barb: } Q\Downarrow. a\]
\[\text{shows P\Downarrow. a}\]
\[\text{using reflection rel barb}\]
\[\text{by (simp add: STCalWB-def)}\]

\[\text{lemma respection-of-barbs-in-barbed-encoding}\]
\[\text{fixes Rel :: (procS, procT) Proc} \times (procS, procT) \text{ Proc}\]
\[\text{and P Q :: (procS, procT) Proc}\]
\[\text{and a :: 'barbs}\]
\[\text{assumes respection: rel-respects-barbs Rel (STCalWB SWB TWB)}\]
\[\text{and rel: } (P, Q) \in \text{Rel}\]
\[\text{shows P\Downarrow. a}\]
\[\text{using preservation-of-barbs-in-barbed-encoding}\]
\[\text{where Rel=Rel and P=P and Q=Q and a=a}\]
\[\text{reflection-of-barbs-in-barbed-encoding}\]
\[\text{where Rel=Rel and P=P and Q=Q and a=a}\]
\[\text{respection rel}\]
\[\text{by blast}\]

A term P of Proc reaches a barb a if either P is a source term that reaches a or P is a target term that reaches a.

\[\text{abbreviation reachesBarbST}\]
\[:: (\text{procS, procT}) \text{ Proc} \Rightarrow \text{ 'barbs} \Rightarrow \text{ bool (-}\Downarrow.- [70, 70] 80)\]
\[\text{where}\]
\[P\Downarrow. a \equiv (\exists S. S \in S P \land S\Downarrow.<SWB> a) \lor (\exists T. T \in T P \land T\Downarrow.<TWB> a)\]

\[\text{lemma STCalWB-reachesBarbST}\]
\[\text{fixes P :: (procS, procT) Proc}\]
\[\text{and a :: 'barbs}\]
\[\text{shows P\Downarrow.<STCalWB SWB TWB>a} = P\Downarrow. a\]
\[\text{proof}\]
\[\text{have } \forall S. \text{SourceTerm S\Downarrow.<STCalWB SWB TWB>a} = \text{SourceTerm S\Downarrow. a}\]
\[\text{using STCal-steps(1)}\]
\[\text{by (auto simp add: STCalWB-def calS calT)}\]
\[\text{moreover have } \forall T. \text{TargetTerm T\Downarrow.<STCalWB SWB TWB>a} = \text{TargetTerm T\Downarrow. a}\]
\[\text{using STCal-steps(2)}\]
\[\text{by (auto simp add: STCalWB-def calS calT)}\]
\[\text{ultimately show } P\Downarrow.<STCalWB SWB TWB>a = P\Downarrow. a\]
\[\text{by (cases P, simp+)}\]
\[\text{qed}\]

\[\text{lemma weak-preservation-of-barbs-in-barbed-encoding}\]
\[\text{fixes Rel :: (procS, procT) Proc} \times (procS, procT) \text{ Proc}\]
\[\text{and P Q :: (procS, procT) Proc}\]
\[\text{and a :: 'barbs}\]
\[\text{assumes preservation: rel-weakly-preserves-barbs Rel (STCalWB SWB TWB)}\]
\[\text{and rel: } (P, Q) \in \text{Rel}\]
\[\text{and barb: } P\Downarrow. a\]
\[\text{shows Q\Downarrow. a}\]
\[\text{proof}\]
\[\text{from barb have } P\Downarrow.<STCalWB SWB TWB>a\]
\[\text{by (simp add: STCalWB-reachesBarbST)}\]
\[\text{with preservation rel have } Q\Downarrow.<STCalWB SWB TWB>a\]
by blast
thus Q ♦. a
by (simp add: STCalWB-reachesBarbST)

done

lemma weak-reflection-of-barbs-in-barbed-encoding:
fixes Rel :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) set
and P Q :: ('procS, 'procT) Proc
and a :: 'barbs
assumes reflection: rel-weakly-reflects-barbs Rel (STCalWB SWB TWB)
and rel: (P, Q) ∈ Rel
and barb: Q ♦. a
shows P ♦. a

proof
from barb have Q ♦<STCalWB SWB TWB>a
by (simp add: STCalWB-reachesBarbST)
with reflection rel have P ♦<STCalWB SWB TWB>a
by blast
thus P ♦. a
by (simp add: STCalWB-reachesBarbST)
done

lemma weak-respection-of-barbs-in-barbed-encoding:
fixes Rel :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) set
and P Q :: ('procS, 'procT) Proc
and a :: 'barbs
assumes respection: rel-weakly-respects-barbs Rel (STCalWB SWB TWB)
and rel: (P, Q) ∈ Rel
shows P ♦. a = Q ♦. a

proof (rule iffI)
assume P ♦. a
with respection rel show Q ♦. a
using weak-preservation-of-barbs-in-barbed-encoding[where Rel=Rel]
by blast

next
assume Q ♦. a
with respection rel show P ♦. a
using weak-reflection-of-barbs-in-barbed-encoding[where Rel=Rel]
by blast

qed

end

theory SourceTargetRelation
imports Encodings SimulationRelations
begin

5 Relation between Source and Target Terms

5.1 Relations Induced by the Encoding Function

We map encodability criteria on conditions of relations between source and target terms. The encoding
function itself induces such relations. To analyse the preservation of source term behaviours we use
relations that contain the pairs (S, enc S) for all source terms S.

inductive-set (in encoding) indRelR
:: (((procS, 'procT) Proc × (procS, 'procT) Proc) set
where
encR: (SourceTerm S, TargetTerm ([S])) ∈ indRelR

54
abbreviation (in encoding) indRelRInfix ::
('procS', 'procT) Proc ⇒ ('procS', 'procT) Proc ⇒ bool (- R·] R - [75, 75] 80)
where
P R·] R Q ≡ (P, Q) ∈ indRelR

inductive-set (in encoding) indRelRPO
:: ((('procS', 'procT) Proc) × (('procS, 'procT) Proc) set
where
encR: (SourceTerm S, TargetTerm ([S])) ∈ indRelRPO |
source: (SourceTerm S, SourceTerm S) ∈ indRelRPO |
target: (TargetTerm T, TargetTerm T) ∈ indRelRPO |
trans: [(P, Q) ∈ indRelRPO; (Q, R) ∈ indRelRPO] ⇒ (P, R) ∈ indRelRPO

abbreviation (in encoding) indRelRPOinfix ::
('procS', 'procT) Proc ⇒ ('procS, 'procT) Proc ⇒ bool (- ≲·] R - [75, 75] 80)
where
P ≲·] R Q ≡ (P, Q) ∈ indRelRPO

lemma (in encoding) indRelRPO-refl:
shows refl indRelRPO
unfolding refl-on-def
proof auto
fix P
show P ≤·] R P
proof (cases P)
case (SourceTerm SP)
assume SP ∈ S P
thus P ≤·] R P
by (simp add: indRelRPO.source)
next
case (TargetTerm TP)
assume TP ∈ T P
thus P ≤·] R P
by (simp add: indRelRPO.target)
qed
qed

lemma (in encoding) indRelRPO-is-preorder:
shows preorder indRelRPO
unfolding preorder-on-def
proof
show refl indRelRPO
by (rule indRelRPO-refl)
next
show trans indRelRPO
unfolding trans-def
proof clarify
fix P Q R
assume P ≤·] R Q and Q ≤·] R R
thus P ≤·] R R
by (rule indRelRPO.trans)
qed
qed

lemma (in encoding) refl-trans-closure-of-indRelR:
shows indRelRPO = indRelR∗
proof auto
fix P Q
assume P ≤·] R Q
thus (P, Q) ∈ indRelR∗
proof induct
case (encR S)
  show (SourceTerm S, TargetTerm ([S])) ∈ indRelR∗
    using indRelR.encR[of S]
    by simp
next
case (source S)
  show (SourceTerm S, SourceTerm S) ∈ indRelR∗
    by simp
next
case (target T)
  show (TargetTerm T, TargetTerm T) ∈ indRelR∗
    by simp
next
case (trans P Q R)
  assume (P, Q) ∈ indRelR∗ and (Q, R) ∈ indRelR∗
  thus (P, R) ∈ indRelR∗
    by simp
qed

next
fix P Q
assume (P, Q) ∈ indRelR∗
thus P ≲ R Q
proof induct
  show P ≲ R P
    using indRelRPO-refl
    unfolding refl-on-def
    by simp
next
  case (step Q R)
    assume P ≲ R Q
    moreover assume Q R ≲ R R
    hence Q ≲ R R
      by (induct, simp add: indRelRPO.encR)
    ultimately show P ≲ R R
      by (rule indRelRPO.trans)
  qed
qed

The relation indRelR is the smallest relation that relates all source terms and their literal translations. Thus there exists a relation that relates source terms and their literal translations and satisfies some predicate on its pairs if the predicate holds for the pairs of indRelR.

lemma (in encoding) indRelR-impl-exists-source-target-relation:
  fixes PredA :: (((procS, 'procT) Proc × (procS, 'procT) Proc) set ⇒ bool
  and PredB :: (((procS, 'procT) Proc × (procS, 'procT) Proc ⇒ bool
  shows PredA indRelR ⇒ ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ PredA Rel
  and (∀ (P, Q) ∈ indRelR. PredB (P, Q))
    ⇒ ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ (∀ (P, Q) ∈ Rel. PredB (P, Q))
proof
  have A: ∀ S. SourceTerm S R R TargetTerm ([S])
    by (simp add: indRelR.encR)
  thus PredA indRelR ⇒ ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ PredA Rel
    by blast
  with A show ∀ (P, Q) ∈ indRelR. PredB (P, Q)
    ⇒ ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ (∀ (P, Q) ∈ Rel. PredB (P, Q))
    by blast
  qed

lemma (in encoding) source-target-relation-impl-indRelR:
  fixes Rel :: (((procS, 'procT) Proc × (procS, 'procT) Proc) set
and Pred :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) ⇒ bool
assumes encRRel: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
and condRel: ∀ (P, Q) ∈ Rel. Pred (P, Q)
s shows ∀ (P, Q) ∈ indRelR. Pred (P, Q)
proof clarify
fix P Q
assume P \ R[\_ ]R Q
with encRRel have (P, Q) ∈ Rel
  by (auto simp add: indRelR.encR)
with condRel show Pred (P, Q)
  by simp
qed

lemma (in encoding) indRelR-iff-exists-source-target-relation:
fixes Pred :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) ⇒ bool
shows (\forall (P, Q) ∈ indRelR. Pred (P, Q))
= (\exists Rel. (\forall S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ (\forall (P, Q) ∈ Rel. Pred (P, Q))
using indRelR-impl-exists-source-target-relation(2)[where PredB=Pred]
  source-target-relation-impl-indRelR[where Pred=Pred]
  by blast

lemma (in encoding) indRelR-modulo-pred-impl-indRelRPO-modulo-pred:
fixes Pred :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) ⇒ bool
assumes reflCond: ∀ P. Pred (P, P)
  and transCond: ∀ P Q R. Pred (P, Q) ∧ Pred (Q, P) → Pred (P, R)
s shows (\forall (P, Q) ∈ indRelR. Pred (P, Q)) = (\forall (P, Q) ∈ indRelRPO. Pred (P, Q))
proof auto
fix P Q
assume A: ∀ x ∈ indRelR. Pred x
assume P ≤_\_R R Q
thus Pred (P, Q)
proof induct
case (encR S)
  have SourceTerm S \ R[\_ ]R TargetTerm ([S])
    by (simp add: indRelR.encR)
  with A show Pred (SourceTerm S, TargetTerm ([S]))
    by simp
next
case (source S)
  from reflCond show Pred (SourceTerm S, SourceTerm S)
    by simp
next
case (target T)
  from reflCond show Pred (TargetTerm T, TargetTerm T)
    by simp
next
case (trans P Q R)
  assume Pred (P, Q) and Pred (Q, R)
  with transCond show Pred (P, R)
    by blast
qed
next
fix P Q
assume \forall x ∈ indRelRPO. Pred x and P \ R[\_ ]R Q
thus Pred (P, Q)
  by (auto simp add: indRelRPO.encR indRelR.simps)
qed

lemma (in encoding) indRelRPO-iff-exists-source-target-relation:
fixes Pred :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) ⇒ bool
shows (\forall (P, Q) ∈ indRelRPO. Pred (P, Q)) = (\exists Rel. (\forall S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
Lemma (in encoding) enc-satisfies-pred-impl-indRelR-satisfies-pred:

fixes \( P \) :: \((\text{procS}, \text{procT}) \times (\text{procS}, \text{procT})\).
assumes \( \text{encCond} : \forall S. \text{Pred} (\text{SourceTerm} S, \text{TargetTerm} ([S])) \)
shows \( \forall (P, Q) \in \text{indRelR}. \text{Pred} (P, Q) \)
  by (auto simp add: \text{encCond} \text{indRelR}.\text{.simps})

Lemma (in encoding) indRelR-satisfies-pred-impl-enc-satisfies-pred:

fixes \( P \) :: \((\text{procS}, \text{procT}) \times (\text{procS}, \text{procT})\).
assumes \( \text{relCond} : \forall (P, Q) \in \text{indRelR}. \text{Pred} (P, Q) \)
shows \( \forall S. \text{Pred} (\text{SourceTerm} S, \text{TargetTerm} ([S])) \)

An encoding preserves, reflects, or respects a predicate iff \( \text{indRelR} \) preserves, reflects, or respects this predicate.
```
using relCond indRelR.encR
by simp

lemma (in encoding) enc-satisfies-pred-iff-indRelR-satisfies-pred:
  fixes Pred :: (('procS', 'procT) Proc × ('procS', 'procT) Proc) ⇒ bool
  shows (∀ S. Pred (SourceTerm S, TargetTerm ([S])) = (∀ (P, Q) ∈ indRelR. Pred (P, Q))
  using enc-satisfies-pred-impl-indRelR-satisfies-pred[where Pred=Pred]
  indRelR-satisfies-pred-impl-enc-satisfies-pred[where Pred=Pred]
by blast

lemma (in encoding) enc-satisfies-binary-pred-iff-indRelR-satisfies-binary-pred:
  fixes Pred :: (('procS', 'procT) Proc × ('procS', 'procT) Proc) ⇒ 'b ⇒ bool
  shows enc-satisfies-binary-pred Pred = rel-satisfies-binary-pred indRelR Pred
  using enc-satisfies-binary-pred-iff-indRelR-satisfies-binary-pred[where Pred=λ(P, Q). Pred P → Pred Q]
by simp

lemma (in encoding) enc-preserves-pred-iff-indRelR-preserves-pred:
  fixes Pred :: ('procS', 'procT) Proc ⇒ bool
  shows enc-preserves-pred Pred = rel-preserves-pred indRelR Pred
  using enc-satisfies-pred-iff-indRelR-satisfies-pred
by blast

lemma (in encoding) enc-preserves-binary-pred-iff-indRelR-preserves-binary-pred:
  fixes Pred :: ('procS', 'procT) Proc ⇒ 'b ⇒ bool
  shows enc-preserves-binary-pred Pred = rel-preserves-binary-pred indRelR Pred
  using enc-satisfies-binary-pred-iff-indRelR-satisfies-binary-pred[where Pred=λ(P, Q) a. Pred P a → Pred Q a]
by blast

lemma (in encoding) enc-preserves-pred-iff-indRelRPO-preserves-pred:
  fixes Pred :: ('procS', 'procT) Proc ⇒ bool
  shows enc-preserves-pred Pred = rel-preserves-pred indRelRPO Pred
  using enc-preserves-pred-iff-indRelR-preserves-pred[where Pred=Pred]
  indRelRPO-module-pred-impl-indRelRPO-module-pred[where Pred=λ(P, Q). Pred P → Pred Q]
by blast

lemma (in encoding) enc-reflects-pred-iff-indRelR-reflects-pred:
  fixes Pred :: ('procS', 'procT) Proc ⇒ bool
  shows enc-reflects-pred Pred = rel-reflects-pred indRelR Pred
  using enc-satisfies-pred-iff-indRelR-satisfies-pred[where Pred=λ(P, Q). Pred Q → Pred P]
by blast

lemma (in encoding) enc-reflects-binary-pred-iff-indRelR-reflects-binary-pred:
  fixes Pred :: ('procS', 'procT) Proc ⇒ 'b ⇒ bool
  shows enc-reflects-binary-pred Pred = rel-reflects-binary-pred indRelR Pred
  using enc-satisfies-binary-pred-iff-indRelR-satisfies-binary-pred[where Pred=λ(P, Q) a. Pred Q a → Pred P a]
by blast

lemma (in encoding) enc-reflects-pred-iff-indRelRPO-reflects-pred:
  fixes Pred :: ('procS', 'procT) Proc ⇒ bool
  shows enc-reflects-pred Pred = rel-reflects-pred indRelRPO Pred
  using enc-reflects-binary-pred-iff-indRelR-reflects-binary-pred[where Pred=Pred]
  indRelRPO-module-pred-impl-indRelRPO-module-pred[where Pred=λ(P, Q). Pred Q → Pred P]
by blast

lemma (in encoding) enc-respects-pred-iff-indRelR-respects-pred:
  fixes Pred :: ('procS', 'procT) Proc ⇒ bool
  shows enc-respects-pred Pred = rel-respects-pred indRelR Pred
```
using enc-preserves-pred-if-indRelR-preserves-pred[where Pred=Pred]
enc-reflects-pred-if-indRelR-reflects-pred[where Pred=Pred]

by blast

lemma (in encoding) enc-respects-binary-pred-if-indRelR-respects-binary-pred:
fixes Pred :: ('procS, 'procT) Proc ⇒ bool
shows enc-respects-binary-pred Pred = rel-respects-binary-pred indRelR Pred
  using enc-preserves-pred-if-indRelR-preserves-binary-pred[where Pred=Pred]
  enc-reflects-binary-pred-if-indRelR-reflects-binary-pred[where Pred=Pred]
  by blast

lemma (in encoding) enc-respects-pred-if-indRelRPO-respects-pred:
fixes Pred :: ('procS, 'procT) Proc ⇒ bool
shows enc-respects-pred Pred = rel-respects-pred indRelRPO Pred
  using enc-preserves-pred-if-indRelR-preserves-binary-pred[where Pred=Pred]
  indRelR-modulo-pred-impl-indRelRPO-modulo-pred[where Pred=λ(P, Q). Pred Q = Pred P]
  apply simp by blast

Accordingly an encoding preserves, reflects, or respects a predicate iff there exists a relation that relates source terms with their literal translations and preserves, reflects, or respects this predicate.

lemma (in encoding) enc-satisfies-pred-if-indRelR-satisfies-pred:
fixes Pred :: ('procS, 'procT) Proc ⇒ bool
shows (∀ S. Pred (SourceTerm S, TargetTerm ([S])))
  = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S]))) ∈ Rel) ∧ (∀ (P, Q) ∈ Rel. Pred (P, Q))
  and (∀ P Q R. Pred (P, Q) ∧ Pred (Q, R) ⟹ Pred (P, R); ∀ P. Pred (P, P))]
  ⟹ (∀ S. Pred (SourceTerm S, TargetTerm ([S])))
  = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S]))) ∈ Rel) ∧ (∀ (P, Q) ∈ Rel. Pred (P, Q)) ∧ preorder Rel
proof –
  show (∀ S. Pred (SourceTerm S, TargetTerm ([S])))
  = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S]))) ∈ Rel) ∧ (∀ (P, Q) ∈ Rel. Pred (P, Q))
  using enc-satisfies-pred-if-indRelR-satisfies-pred[where Pred=Pred]
  indRelR-iff-exists-source-target-relation[where Pred=Pred]
  by simp
next
  have (∀ S. Pred (SourceTerm S, TargetTerm ([S]))) = (∀ (P, Q) ∈ indRelR. Pred (P, Q))
  using enc-satisfies-pred-if-indRelR-satisfies-pred[where Pred=Pred]
  by simp
moreover assume ∀ P Q R. Pred (P, Q) ∧ Pred (Q, R) ⟹ Pred (P, R) and ∀ P. Pred (P, P)
  hence (∀ (P, Q) ∈ indRelR. Pred (P, Q)) = (∀ (P, Q) ∈ indRelRPO. Pred (P, Q))
  using indRelR-modulo-pred-impl-indRelRPO-modulo-pred[where Pred=Pred]
  by blast
ultimately show (∀ S. Pred (SourceTerm S, TargetTerm ([S]))) = (∃ Rel.
  (∀ S. (SourceTerm S, TargetTerm ([S]))) ∈ Rel) ∧ (∀ (P, Q) ∈ Rel. Pred (P, Q)) ∧ preorder Rel)
  using indRelRPO-iff-exists-source-target-relation[where Pred=Pred]
  by simp
qed

lemma (in encoding) enc-preserves-pred-if-source-target-rel-preserves-pred:
fixes Pred :: ('procS, 'procT) Proc ⇒ bool
shows enc-preserves-pred Pred
  = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S]))) ∈ Rel) ∧ rel-preserves-pred Pred Rel Pred)
and enc-preserves-pred Pred = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S]))) ∈ Rel)
  ∧ rel-preserves-pred Pred Rel Pred ∧ preorder Rel)
proof –
  have A1: enc-preserves-pred Pred
    = (∀ S. (λ(P, Q). Pred P ⟹ Pred Q) (SourceTerm S, TargetTerm ([S])))
    by blast
moreover have A2: Pred Rel Pred
  = (∀ (P, Q) ∈ Rel. (λ(P, Q). Pred P ⟹ Pred Q) (P, Q))
  by blast
ultimately show \( \text{enc-preserves-pred} \) \( \text{Pred} = (\exists \text{Rel}. (\forall \text{S}. (\text{SourceTerm} \text{S}, \text{TargetTerm} ([\text{S}])) \in \text{Rel}) \land \text{rel-preserves-pred Rel} \text{Pred}) \)

using \( \text{enc-satisfies-pred-iff-source-target-satisfies-pred} \) \( (1) \) \( \text{where} \)
\[ \text{Pred} = \lambda (P, Q). \text{Pred} P \rightarrow \text{Pred} Q \]
by simp

from \( A1 \) \( A2 \) show \( \text{enc-preserves-pred} \) \( \text{Pred} = (\exists \text{Rel}. (\forall \text{S}. (\text{SourceTerm} \text{S}, \text{TargetTerm} ([\text{S}])) \in \text{Rel}) \land \text{rel-preserves-pred Rel} \text{Pred} \land \text{preorder Rel}) \)

using \( \text{enc-satisfies-pred-iff-source-target-satisfies-pred} \) \( (2) \) \( \text{where} \)
\[ \text{Pred} = \lambda (P, Q). \forall a. \text{Pred} P a \rightarrow \text{Pred} Q a \]
by simp
qed

lemma (in encoding) \( \text{enc-preserves-binary-pred-iff-source-target-rel-preserves-binary-pred} \):

fixes \( \text{Pred} :: (\langle \text{procS}, \text{procT} \rangle \text{Proc} \Rightarrow \text{bool} \)

shows \( \text{enc-preserves-binary-pred} \) \( \text{Pred} = (\exists \text{Rel}. (\forall \text{S}. (\text{SourceTerm} \text{S}, \text{TargetTerm} ([\text{S}])) \in \text{Rel}) \land \text{rel-preserves-binary-pred Rel} \text{Pred}) \)

proof –

have \( \text{enc-preserves-binary-pred} \) \( \text{Pred} \)
\[ = (\forall \text{S}. (\lambda (P, Q)). \forall a. \text{Pred} P a \rightarrow \text{Pred} Q a) (\text{SourceTerm} \text{S}, \text{TargetTerm} ([\text{S}])) \]
by blast

moreover have \( \text{Rel} \land \text{rel-preserves-binary-pred Rel} \text{Pred} \)
\[ = (\forall (P, Q) \in \text{Rel}. (\lambda (P, Q)). \forall a. \text{Pred} P a \rightarrow \text{Pred} Q a) (P, Q) \]
by blast

ultimately show \( \text{enc-preserves-binary-pred} \) \( \text{Pred} = (\exists \text{Rel}. (\forall \text{S}. (\text{SourceTerm} \text{S}, \text{TargetTerm} ([\text{S}])) \in \text{Rel}) \land \text{rel-preserves-binary-pred Rel} \text{Pred}) \)

using \( \text{enc-satisfies-pred-iff-source-target-satisfies-pred} \) \( (1) \) \( \text{where} \)
\[ \text{Pred} = \lambda (P, Q). \forall a. \text{Pred} P a \rightarrow \text{Pred} Q a \]
by simp
qed

lemma (in encoding) \( \text{enc-reflects-binary-pred-iff-source-target-rel-reflects-binary-pred} \):

fixes \( \text{Pred} :: (\langle \text{procS}, \text{procT} \rangle \text{Proc} \Rightarrow \text{bool} \)

shows \( \text{enc-reflects-binary-pred} \) \( \text{Pred} = (\exists \text{Rel}. (\forall \text{S}. (\text{SourceTerm} \text{S}, \text{TargetTerm} ([\text{S}])) \in \text{Rel}) \land \text{rel-reflects-binary-pred Rel} \text{Pred}) \)

proof –

have \( \text{A1: enc-reflects-binary-pred} \) \( \text{Pred} \)
\[ = (\forall \text{S}. (\lambda (P, Q)). \text{Pred} Q \rightarrow \text{Pred} P) (\text{SourceTerm} \text{S}, \text{TargetTerm} ([\text{S}])) \]
by blast

moreover have \( \text{A2: rel-reflects-binary-pred Rel} \text{Pred} \)
\[ = (\forall (P, Q) \in \text{Rel}. (\lambda (P, Q)). \text{Pred} Q \rightarrow \text{Pred} P) (P, Q) \]
by blast

ultimately show \( \text{enc-reflects-binary-pred} \) \( \text{Pred} = (\exists \text{Rel}. (\forall \text{S}. (\text{SourceTerm} \text{S}, \text{TargetTerm} ([\text{S}])) \in \text{Rel}) \land \text{rel-reflects-binary-pred Rel} \text{Pred}) \)

using \( \text{enc-satisfies-pred-iff-source-target-satisfies-pred} \) \( (1) \) \( \text{where} \)
\[ \text{Pred} = \lambda (P, Q). \forall a. \text{Pred} P a \rightarrow \text{Pred} Q a \]
by simp

from \( A1 \) \( A2 \) show \( \text{enc-reflects-binary-pred} \) \( \text{Pred} = (\exists \text{Rel}. (\forall \text{S}. (\text{SourceTerm} \text{S}, \text{TargetTerm} ([\text{S}])) \in \text{Rel}) \land \text{rel-reflects-binary-pred Rel} \text{Pred}) \)

using \( \text{enc-satisfies-pred-iff-source-target-satisfies-pred} \) \( (2) \) \( \text{where} \)
\[ \text{Pred} = \lambda (P, Q). \forall a. \text{Pred} P a \rightarrow \text{Pred} Q a \]
by simp
qed

lemma (in encoding) \( \text{enc-reflects-binary-pred-iff-source-target-rel-reflects-binary-pred} \):

fixes \( \text{Pred} :: (\langle \text{procS}, \text{procT} \rangle \text{Proc} \Rightarrow \text{bool} \)

shows \( \text{enc-reflects-binary-pred} \) \( \text{Pred} = (\exists \text{Rel}. (\forall \text{S}. (\text{SourceTerm} \text{S}, \text{TargetTerm} ([\text{S}])) \in \text{Rel}) \land \text{rel-reflects-binary-pred Rel} \text{Pred}) \)

proof –

have \( \text{enc-reflects-binary-pred} \) \( \text{Pred} \)
inductive-set for all source terms S.

To analyse the reflection of source term behaviours we use relations that contain the pairs (enc S, S)

moreover have \( \forall (P, Q) \in \text{Rel}. \ (\lambda(P, Q). \forall a. \ \text{Pred} P a \rightarrow \text{Pred} P a) \ (P, Q) \)

by blast

ultimately show \( \text{enc-reflects-binary-pred} \ \text{Pred} = (\exists \text{Rel}. \ (\forall S. \ (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}) \land \text{rel-reflects-binary-pred} \ \text{Rel} \ \text{Pred}) \)

using \( \text{enc-satisfies-pred-iff-source-target-satisfies-pred} (1) \) [where
\( \text{Pred} = \lambda(P, Q). \forall a. \ \text{Pred} P a \rightarrow \text{Pred} P a \)]

by simp

qed

\text{lemma (in encoding)} \ \text{enc-respects-pred-iff-source-target-rel-respects-pred-encR:}

\text{fixes} \ \text{Pred} :: ('procS, 'procT) \text{Proc} \Rightarrow \text{bool}

\text{shows} \ \text{enc-respects-pred Pred}

= (\exists \text{Rel}. \ (\forall S. \ (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}) \land \text{rel-respects-pred} \ \text{Rel} \ \text{Pred})

and \( \text{enc-respects-pred Pred} = (\exists \text{Rel}. \ (\forall S. \ (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}) \land \text{rel-respects-pred} \ \text{Rel} \ \text{Pred} \land \text{preorder} \ \text{Rel}) \)

\text{proof}

have A1: \( \text{enc-respects-pred Pred} = (\forall S. \ (\lambda(P, Q). \ \text{Pred} P = \text{Pred} Q) \ (\text{SourceTerm} S, \text{TargetTerm} ([S]))) \)

by blast

moreover have A2: \( \forall (P, Q) \in \text{Rel}. \ (\lambda(P, Q). \ \text{Pred} P = \text{Pred} Q) \ (P, Q) \)

by blast

ultimately show \( \text{enc-respects-pred Pred} = (\exists \text{Rel}. \ (\forall S. \ (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}) \land \text{rel-respects-pred} \ \text{Rel} \ \text{Pred} \land \text{preorder} \ \text{Rel}) \)

using \( \text{enc-satisfies-pred-iff-source-target-satisfies-pred} (2) \) [where
\( \text{Pred} = \lambda(P, Q). \forall a. \ \text{Pred} P a = \text{Pred} P a \)]

by simp

qed

\text{lemma (in encoding)} \ \text{enc-respects-binary-pred-iff-source-target-rel-respects-binary-pred-encR:}

\text{fixes} \ \text{Pred} :: ('procS, 'procT) \text{Proc} \Rightarrow \text{bool}

\text{shows} \ \text{enc-respects-binary-pred Pred} = (\exists \text{Rel}. \ (\forall S. \ (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}) \land \text{rel-respects-binary-pred} \ \text{Rel} \ \text{Pred})

\text{proof}

have \( \text{enc-respects-binary-pred Pred} = (\forall S. \ (\lambda(P, Q). \ \text{Pred} P a = \text{Pred} Q a) \ (\text{SourceTerm} S, \text{TargetTerm} ([S]))) \)

by blast

moreover have \( \forall (P, Q) \in \text{Rel}. \ (\lambda(P, Q). \ \text{Pred} P a = \text{Pred} Q a) \ (P, Q) \)

by blast

ultimately show \( \text{enc-respects-binary-pred Pred} = (\exists \text{Rel}. \ (\forall S. \ (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}) \land \text{rel-respects-binary-pred} \ \text{Rel} \ \text{Pred}) \)

using \( \text{enc-satisfies-pred-iff-source-target-satisfies-pred} (1) \) [where
\( \text{Pred} = \lambda(P, Q). \forall a. \ \text{Pred} P a = \text{Pred} Q a \)]

by simp

qed

To analyse the reflection of source term behaviours we use relations that contain the pairs (enc S, S)
for all source terms S.

inductive-set (in encoding) \( \text{indRelL} \)

\( :: (((\text{procS}, \text{procT}) \text{Proc}) \times ((\text{procS}, \text{procT}) \text{Proc})) \text{set} \)

where
encL: (TargetTerm ([S]), SourceTerm S) ∈ indRelL

abbreviation (in encoding) indRelLInfix ::
  ('procS', 'procT) Proc ⇒ ('procS', 'procT) Proc ⇒ bool (- R[]L - [75, 75] 80)
  where
  P R[]L Q ≡ (P, Q) ∈ indRelL

inductive-set (in encoding) indRelLPO
  :: ((('procS', 'procT) Proc) × ('procS', 'procT) Proc) set
  where
  encL: (TargetTerm ([S]), SourceTerm S) ∈ indRelLPO |
  source: (TargetTerm S, SourceTerm S) ∈ indRelLPO |
  target: (TargetTerm T, TargetTerm T) ∈ indRelLPO |
  trans: [(P, Q) ∈ indRelLPO; (Q, R) ∈ indRelLPO] ⇒ (P, R) ∈ indRelLPO

abbreviation (in encoding) indRelLPOInfix ::
  ('procS', 'procT) Proc ⇒ ('procS', 'procT) Proc ⇒ bool (- ≲[]L - [75, 75] 80)
  where
  P ≲[]L Q ≡ (P, Q) ∈ indRelLPO

lemma (in encoding) indRelLPO-refl:
  shows refl indRelLPO
  unfolding refl-on-def
proof auto
  fix P
  show P ≲[]L P
  proof (cases P)
    case (SourceTerm SP)
      assume SP ∈ S P
      thus P ≲[]L P
        by (simp add: indRelLPO.source)
  next
    case (TargetTerm TP)
      assume TP ∈ T P
      thus P ≲[]L P
        by (simp add: indRelLPO.target)
  qed
qed

lemma (in encoding) indRelLPO-is-preorder:
  shows preorder indRelLPO
  unfolding preorder-on-def
proof
  show refl indRelLPO
    by (rule indRelLPO-refl)
next
  show trans indRelLPO
    unfolding trans-def
  proof clarify
    fix P Q R
    assume P ≲[]L Q and Q ≲[]L R
    thus P ≲[]L R
      by (rule indRelLPO.trans)
  qed
qed

lemma (in encoding) refl-trans-closure-of-indRelL:
  shows indRelLPO = indRelL*
proof auto
  fix P Q
  assume P ≲[]L Q
thus \((P, Q) \in \text{indRelL}^*\)

proof
  induct
  case (\text{encL} S)
    show \((\text{TargetTerm} ([S]), \text{SourceTerm} S) \in \text{indRelL}^*\)
      using \text{indRelL}.\text{encL}[\text{of} S]
      by simp
  next
  case (\text{source} S)
    show \((\text{SourceTerm} S, \text{SourceTerm} S) \in \text{indRelL}^*\)
      by simp
  next
  case (\text{target} T)
    show \((\text{TargetTerm} T, \text{TargetTerm} T) \in \text{indRelL}^*\)
      by simp
  next
  case (\text{trans} P Q R)
    assume \((P, Q) \in \text{indRelL}^*\) and \((Q, R) \in \text{indRelL}^*\)
    thus \((P, R) \in \text{indRelL}^*\)
    by simp
  qed
  next
  fix \(P\) \(Q\)
  assume \((P, Q) \in \text{indRelL}^*\)
  thus \(P \preceq \bigl[\bigl[\cdot \bigr]\bigr] L Q\)
  proof
    induct
    show \(P \preceq \bigl[\bigl[\cdot \bigr]\bigr] L P\)
      using \text{indRelL}.\text{encL}.\text{refl}
      unfolding \text{refl-on-def}
      by simp
    next
    case (\text{step} Q R)
    assume \(P \preceq \bigl[\bigl[\cdot \bigr]\bigr] L Q\)
    moreover assume \(Q \preceq \bigl[\bigl[\cdot \bigr]\bigr] L R\)
    hence \(Q \preceq \bigl[\bigl[\cdot \bigr]\bigr] R P\)
      by (induct, simp add: \text{indRelL}.\text{encL}.\text{lpo-refl})
    ultimately show \(P \preceq \bigl[\bigl[\cdot \bigr]\bigr] L R\)
      by (simp add: \text{indRelL}.\text{lpo-refl}.\text{trans}[\text{of} P Q R])
  qed
  qed

The relations \text{indRelR} and \text{indRelL} are dual. \text{indRelR} preserves some predicate iff \text{indRelL} reflects it. \text{indRelR} reflects some predicate iff \text{indRelL} reflects it. \text{indRelR} respects some predicate iff \text{indRelL} does.

\textbf{lemma (in encoding) \text{indRelR}-preserves-pred-iff-indRelL-reflects-pred:}

  \textbf{fixes} \(\text{Pred} :: (\text{'procS}, \text{'procT}) \text{Proc} \Rightarrow \text{bool}\)
  \textbf{shows} \(\text{rel-preserves-pred \text{indRelR} \text{Pred} = \text{rel-reflects-pred \text{indRelL} \text{Pred}}}\)

\textbf{proof}
  assume \(\text{preservation} \)\(\text{indRelR}-\text{preserves-pred} \text{\text{Pred}}\)
  show \(\text{rel-reflects-pred \text{indRelL} \text{\text{Pred}}}\)
  proof
    clarify
    fix \(P\) \(Q\)
    assume \(P \preceq \bigl[\bigl[\cdot \bigr]\bigr] L Q\)
    from \text{this} obtain \(S\) where \(S \in S Q\) and \([S] \in T P\)
      by (induct, blast)
    hence \(Q \preceq \bigl[\bigl[\cdot \bigr]\bigr] R P\)
      by (simp add: \text{indRelR}.\text{encR})
    moreover assume \(\text{Pred} Q\)
    ultimately show \(\text{Pred} P\)
      using \text{preservation}
      by blast
  qed
assume reflection: rel-reflects-pred indRelL Pred
show rel-preserves-pred indRelR Pred
proof clarify
fix P Q
assume P \mathcal{R} Q
from this obtain S where S \in S P and [S] \in T Q
by (induct, blast)
hence Q \mathcal{R} L P
by (simp add: indRelL.encL)
morerover assume Pred P
ultimately show Pred Q
using reflection
by blast
qed

next
assume reflection: rel-reflects-pred indRelL Pred
show rel-preserves-pred indRelR Pred
proof clarify
fix P Q x
assume P \mathcal{R} L Q
from this obtain S where S \in S Q and [S] \in T P
by (induct, blast)
hence Q \mathcal{R} R P
by (simp add: indRelL.encR)
morerover assume Pred P x
ultimately show Pred Q x
using preservation
by blast
qed

lemma (in encoding) indRelR-preserves-binary-pred-iff-indRelL-reflects-binary-pred:
fixes Pred :: ('procS, 'procT) Proc \Rightarrow bool
shows rel-preserves-binary-pred indRelR Pred = rel-reflects-binary-pred indRelL Pred
proof
assume preservation: rel-preserves-binary-pred indRelR Pred
show rel-reflects-binary-pred indRelL Pred
proof clarify
fix P Q x
assume P \mathcal{R} L Q
from this obtain S where S \in S Q and [S] \in T P
by (induct, blast)
hence Q \mathcal{R} R P
by (simp add: indRelL.encR)
morerover assume Pred P x
ultimately show Pred Q x
using preservation
by blast
qed

next
assume reflection: rel-reflects-binary-pred indRelL Pred
show rel-preserves-binary-pred indRelR Pred
proof clarify
fix P Q x
assume P \mathcal{R} R Q
from this obtain S where S \in S P and [S] \in T Q
by (induct, blast)
hence Q \mathcal{R} L P
by (simp add: indRelL.encL)
morerover assume Pred P x
ultimately show Pred Q x
using reflection
by blast
qed

lemma (in encoding) indRelR-reflects-pred-iff-indRelL-preserves-pred:
fixes Pred :: ('procS, 'procT) Proc \Rightarrow bool
shows rel-reflects-pred indRelR Pred = rel-preserves-pred indRelL Pred
proof
assume reflection: rel-reflects-pred indRelR Pred
show rel-preserves-pred indRelL Pred
proof clarify
fix P Q
assume \( P \mathcal{R} \cdot I \cdot L \cdot Q \)

from this obtain \( S \) where \( S \in S \cdot Q \) and \([S] \in T \cdot P \)
  by (induct, blast)

hence \( Q \mathcal{R} \cdot \cdot I \cdot L \cdot P \)
  by (simp add: indRelR.encR)

moreover assume \( \text{Pred} \cdot P \)

ultimately show \( \text{Pred} \cdot Q \)
  using reflection
  by blast

qed

next

assume preservation: rel-preserves-pred indRelL Pred

show rel-reflects-pred indRelR Pred

proof clarify

fix \( P \cdot Q \)

assume \( P \mathcal{R} \cdot \cdot I \cdot R \cdot Q \)

from this obtain \( S \) where \( S \in S \cdot P \) and \([S] \in T \cdot Q \)
  by (induct, blast)

hence \( Q \mathcal{R} \cdot \cdot I \cdot L \cdot P \)
  by (simp add: indRelL.encL)

moreover assume \( \text{Pred} \cdot Q \cdot x \)

ultimately show \( \text{Pred} \cdot P \cdot x \)
  using preservation
  by blast

qed

qed

lemma (in encoding) indRelR-reflects-binary-pred-iff-indRelL-preserves-binary-pred:

fixes \( \text{Pred} :: ('procS, 'procT) \cdot \text{Proc} \Rightarrow 'b \Rightarrow \text{bool} \)

shows rel-reflects-binary-pred indRelR Pred = rel-preserves-binary-pred indRelL Pred

proof

assume reflection: rel-reflects-binary-pred indRelR Pred

show rel-preserves-binary-pred indRelL Pred

proof clarify

fix \( P \cdot Q \cdot x \)

assume \( P \mathcal{R} \cdot \cdot I \cdot L \cdot Q \)

from this obtain \( S \) where \( S \in S \cdot Q \) and \([S] \in T \cdot P \)
  by (induct, blast)

hence \( Q \mathcal{R} \cdot \cdot I \cdot L \cdot P \)
  by (simp add: indRelL.encL)

moreover assume \( \text{Pred} \cdot P \cdot x \)

ultimately show \( \text{Pred} \cdot Q \cdot x \)
  using reflection
  by blast

qed

next

assume preservation: rel-preserves-binary-pred indRelL Pred

show rel-reflects-binary-pred indRelR Pred

proof clarify

fix \( P \cdot Q \cdot x \)

assume \( P \mathcal{R} \cdot \cdot I \cdot R \cdot Q \)

from this obtain \( S \) where \( S \in S \cdot P \) and \([S] \in T \cdot Q \)
  by (induct, blast)

hence \( Q \mathcal{R} \cdot \cdot I \cdot L \cdot P \)
  by (simp add: indRelL.encL)

moreover assume \( \text{Pred} \cdot Q \cdot x \)

ultimately show \( \text{Pred} \cdot P \cdot x \)
  using preservation
  by blast

qed
lemma (in encoding) \(\text{indRelR-respects-pred-iff-indRelL-respects-pred}\):

\[\begin{align*}
\text{fixes} & \quad \text{Pred} :: ('\text{procS}', '\text{procT}) \text{ Proc} \Rightarrow \text{bool} \\
\text{shows} & \quad \text{rel-respects-pred indRelR Pred} = \text{rel-respects-pred indRelL Pred} \\
\text{using} & \quad \text{indRelR-preserves-pred-iff-indRelL-reflects-pred[where Pred=Pred]} \\
& \quad \text{indRelR-reflects-pred-iff-indRelL-preserves-pred[where Pred=Pred]} \\
\text{by} & \quad \text{blast}
\end{align*}\]

lemma (in encoding) \(\text{indRelR-respects-binary-pred-iff-indRelL-respects-binary-pred}\):

\[\begin{align*}
\text{fixes} & \quad \text{Pred} :: ('\text{procS}', '\text{procT}) \text{ Proc} \Rightarrow \text{bool} \\
\text{shows} & \quad \text{rel-respects-binary-pred indRelR Pred} = \text{rel-respects-binary-pred indRelL Pred} \\
\text{using} & \quad \text{indRelR-preserves-binary-pred-iff-indRelL-reflects-binary-pred[where Pred=Pred]} \\
& \quad \text{indRelR-reflects-binary-pred-iff-indRelL-preserves-binary-pred[where Pred=Pred]} \\
\text{by} & \quad \text{blast}
\end{align*}\]

lemma (in encoding) \(\text{indRelR-cond-preservation-iff-indRelL-cond-reflection}\):

\[\begin{align*}
\text{fixes} & \quad \text{Pred} :: ('\text{procS}', '\text{procT}) \text{ Proc} \Rightarrow \text{bool} \\
\text{shows} & \quad (\exists \text{ Rel}. \forall S. \text{ (SourceTerm S, TargetTerm ([S]]) \in Rel} \wedge \text{rel-preserves-pred Rel Pred}) \\
& \quad = (\exists \text{ Rel}. \forall S. \text{ (TargetTerm ([S]), SourceTerm S) \in Rel} \wedge \text{rel-reflects-pred Rel Pred})
\end{align*}\]

\[\begin{align*}
\text{proof} & \quad \text{assume} \exists \text{ Rel}. \forall S. \text{ (SourceTerm S, TargetTerm ([S])) \in Rel} \wedge \text{rel-preserves-pred Rel Pred} \\
\text{then obtain} & \quad \text{Rel where A1:} \forall S. \text{ (SourceTerm S, TargetTerm ([S])) \in Rel} \\
& \quad \text{and A2:} \text{rel-preserves-pred Rel Pred} \\
& \quad \text{by blast} \\
\text{from A1 have} & \quad \forall S. \text{ (TargetTerm ([S]), SourceTerm S) \in Rel}^{-1} \\
& \quad \text{by simp} \\
\text{moreover from A2 have} & \quad \text{rel-reflects-pred (Rel}^{-1}) \text{ Pred} \\
& \quad \text{by simp} \\
\text{ultimately show} & \quad \exists \text{ Rel}. \forall S. \text{ (TargetTerm ([S]), SourceTerm S) \in Rel} \wedge \text{rel-reflects-pred Rel Pred} \\
& \quad \text{by blast}
\end{align*}\]

\[\begin{align*}
\text{next} & \quad \text{assume} \exists \text{ Rel}. \forall S. \text{ (TargetTerm ([S]), SourceTerm S) \in Rel} \wedge \text{rel-reflects-pred Rel Pred} \\
\text{then obtain} & \quad \text{Rel where B1:} \forall S. \text{ (TargetTerm ([S]), SourceTerm S) \in Rel} \\
& \quad \text{and B2:} \text{rel-reflects-pred Rel Pred} \\
& \quad \text{by blast} \\
\text{from B1 have} & \quad \forall S. \text{ (SourceTerm S, TargetTerm ([S])) \in Rel}^{-1} \\
& \quad \text{by simp} \\
\text{moreover from B2 have} & \quad \text{rel-preserves-pred (Rel}^{-1}) \text{ Pred} \\
& \quad \text{by blast} \\
\text{ultimately} & \quad \text{show} \exists \text{ Rel}. \forall S. \text{ (SourceTerm S, TargetTerm ([S])) \in Rel} \wedge \text{rel-preserves-pred Rel Pred} \\
& \quad \text{by blast}
\end{align*}\]

qed

lemma (in encoding) \(\text{indRelR-cond-binary-preservation-iff-indRelL-cond-binary-reflection}\):

\[\begin{align*}
\text{fixes} & \quad \text{Pred} :: ('\text{procS}', '\text{procT}) \text{ Proc} \Rightarrow 'b \Rightarrow \text{bool} \\
\text{shows} & \quad (\exists \text{ Rel}. \forall S. \text{ (SourceTerm S, TargetTerm ([S])) \in Rel} \wedge \text{rel-preserves-binary-pred Rel Pred}) \\
& \quad = (\exists \text{ Rel}. \forall S. \text{ (TargetTerm ([S]), SourceTerm S) \in Rel} \wedge \text{rel-reflects-binary-pred Rel Pred})
\end{align*}\]

\[\begin{align*}
\text{proof} & \quad \text{assume} \exists \text{ Rel}. \forall S. \text{ (SourceTerm S, TargetTerm ([S])) \in Rel} \wedge \text{rel-preserves-binary-pred Rel Pred} \\
\text{then obtain} & \quad \text{Rel where A1:} \forall S. \text{ (SourceTerm S, TargetTerm ([S])) \in Rel} \\
& \quad \text{and A2:} \text{rel-preserves-binary-pred Rel Pred} \\
& \quad \text{by blast} \\
\text{from A1 have} & \quad \forall S. \text{ (TargetTerm ([S]), SourceTerm S) \in Rel}^{-1} \\
& \quad \text{by simp} \\
\text{moreover from A2 have} & \quad \text{rel-reflects-binary-pred (Rel}^{-1}) \text{ Pred} \\
& \quad \text{by simp} \\
\text{ultimately} & \quad \text{show} \exists \text{ Rel}. \forall S. \text{ (SourceTerm S, TargetTerm ([S])) \in Rel} \wedge \text{rel-reflects-binary-pred Rel Pred} \\
& \quad \text{by blast}
\end{align*}\]
next
assume \( \exists \text{Rel. } (\forall S. \text{ (TargetTerm } (\lfloor S \rfloor), \text{SourceTerm } S) \in \text{Rel} ) \land \text{rel-reflects-binary-pred } \text{Rel Pred} \)
then obtain \( \text{Rel where B1: } (\forall S. \text{ (TargetTerm } (\lfloor S \rfloor), \text{SourceTerm } S) \in \text{Rel} ) \land \text{rel-reflects-binary-pred } \text{Rel Pred} \)
    by blast
from B1 have \( (\forall S. \text{ (SourceTerm } S, \text{TargetTerm } (\lfloor S \rfloor)) \in \text{Rel}^{-1} ) \)
    by simp
moreover from B2 have \( \text{rel-preserves-binary-pred } (\text{Rel}^{-1}) \text{ Pred} \)
    by simp
ultimately show \( (\forall S. \text{ (SourceTerm } S, \text{TargetTerm } (\lfloor S \rfloor)) \in \text{Rel} ) \land \text{rel-preserves-binary-pred } \text{Rel Pred} \)
by blast
qed

lemma \( \text{(in encoding) } \text{indRelR-cond-reflection-iff-indRelL-cond-preservation: } \)
fixes \( \text{Pred} :: (\text{'procS}, \text{'procT}) \text{ Proc } \Rightarrow \text{bool} \)
shows \( (\exists \text{Rel. } (\forall S. \text{ (SourceTerm } S, \text{TargetTerm } (\lfloor S \rfloor)) \in \text{Rel} ) \land \text{rel-reflects-pred } \text{Rel Pred} ) \)
    = \( (\exists \text{Rel. } (\forall S. \text{ (TargetTerm } (\lfloor S \rfloor), \text{SourceTerm } S) \in \text{Rel} ) \land \text{rel-preserves-pred } \text{Rel Pred} ) \)
proof
assume \( (\exists \text{Rel. } (\forall S. \text{ (SourceTerm } S, \text{TargetTerm } (\lfloor S \rfloor)) \in \text{Rel} ) \land \text{rel-reflects-pred } \text{Rel Pred} ) \)
then obtain \( \text{Rel where A1: } (\forall S. \text{ (SourceTerm } S, \text{TargetTerm } (\lfloor S \rfloor)) \in \text{Rel} ) \land \text{rel-reflects-pred } \text{Rel Pred} \)
    by blast
from A1 have \( (\forall S. \text{ (TargetTerm } (\lfloor S \rfloor), \text{SourceTerm } S) \in \text{Rel}^{-1} ) \)
    by simp
moreover from A2 have \( \text{rel-preserves-pred } (\text{Rel}^{-1}) \text{ Pred} \)
    by blast
ultimately show \( (\forall S. \text{ (SourceTerm } S, \text{TargetTerm } (\lfloor S \rfloor)) \in \text{Rel} ) \land \text{rel-preserves-pred } \text{Rel Pred} \)
by blast
next
assume \( (\exists \text{Rel. } (\forall S. \text{ (TargetTerm } (\lfloor S \rfloor), \text{SourceTerm } S) \in \text{Rel} ) \land \text{rel-preserves-pred } \text{Rel Pred} ) \)
then obtain \( \text{Rel where B1: } (\forall S. \text{ (TargetTerm } (\lfloor S \rfloor), \text{SourceTerm } S) \in \text{Rel} ) \land \text{rel-preserves-pred } \text{Rel Pred} \)
    by blast
from B1 have \( (\forall S. \text{ (SourceTerm } S, \text{TargetTerm } (\lfloor S \rfloor)) \in \text{Rel}^{-1} ) \)
    by simp
moreover from B2 have \( \text{rel-reflects-pred } (\text{Rel}^{-1}) \text{ Pred} \)
    by simp
ultimately show \( (\forall S. \text{ (SourceTerm } S, \text{TargetTerm } (\lfloor S \rfloor)) \in \text{Rel} ) \land \text{rel-reflects-pred } \text{Rel Pred} \)
by blast
qed

lemma \( \text{(in encoding) } \text{indRelR-cond-binary-reflection-iff-indRelL-cond-binary-preservation: } \)
fixes \( \text{Pred} :: (\text{procS}, \text{procT}) \text{ Proc } \Rightarrow \text{bool} \)
shows \( (\exists \text{Rel. } (\forall S. \text{ (SourceTerm } S, \text{TargetTerm } (\lfloor S \rfloor)) \in \text{Rel} ) \land \text{rel-reflects-binary-pred } \text{Rel Pred} ) \)
    = \( (\exists \text{Rel. } (\forall S. \text{ (TargetTerm } (\lfloor S \rfloor), \text{SourceTerm } S) \in \text{Rel} ) \land \text{rel-preserves-binary-pred } \text{Rel Pred} ) \land \text{rel-reflects-binary-pred } \text{Rel Pred} \)
proof
assume \( (\exists \text{Rel. } (\forall S. \text{ (SourceTerm } S, \text{TargetTerm } (\lfloor S \rfloor)) \in \text{Rel} ) \land \text{rel-reflects-binary-pred } \text{Rel Pred} ) \)
then obtain \( \text{Rel where A1: } (\forall S. \text{ (SourceTerm } S, \text{TargetTerm } (\lfloor S \rfloor)) \in \text{Rel} ) \land \text{rel-reflects-binary-pred } \text{Rel Pred} \)
    by blast
from A1 have \( (\forall S. \text{ (TargetTerm } (\lfloor S \rfloor), \text{SourceTerm } S) \in \text{Rel}^{-1} ) \)
    by simp
moreover from A2 have \( \text{rel-preserves-binary-pred } (\text{Rel}^{-1}) \text{ Pred} \)
    by blast
ultimately show \( (\forall S. \text{ (SourceTerm } S, \text{TargetTerm } (\lfloor S \rfloor)) \in \text{Rel} ) \land \text{rel-preserves-binary-pred } \text{Rel Pred} \)
by blast
next
asssume ∃ Rel. (∀ S. (TargetTerm ([S]), SourceTerm S) ∈ Rel) ∧ rel-preserves-binary-pred Rel Pred
then obtain Rel where B1: ∀ S. (TargetTerm ([S]), SourceTerm S) ∈ Rel
and B2: rel-preserves-binary-pred Rel Pred

by blast
from B1 have ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel⁻¹
by simp

moreover from B2 have rel-reflects-binary-pred (Rel⁻¹) Pred
by simp

ultimately show ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ rel-reflects-binary-pred Rel Pred
by blast
qed

lemma (in encoding) indRelR-cond-respection-iff-indRelL-cond-respection:
fixes Pred :: ('procS, 'procT) Proc ⇒ bool
shows (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ rel-respects-binary-pred Rel Pred)
= (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ rel-respects-binary-pred Rel Pred)

proof
asssume ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ rel-respects-binary-pred Rel Pred
from this obtain Rel where A1: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
and A2: rel-respects-binary-pred Rel Pred

by blast
from A1 have ∀ S. (TargetTerm ([S]), SourceTerm S) ∈ { (a, b), (b, a) ∈ Rel }
by simp

moreover from A2 have rel-respects-binary-pred { (a, b), (b, a) ∈ Rel } Pred
by blast

ultimately show ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ rel-respects-binary-pred Rel Pred
by blast
next
asssume ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ rel-respects-binary-pred Rel Pred
from this obtain Rel where A1: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
and A2: rel-respects-binary-pred Rel Pred

by blast
from A1 have ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ { (a, b), (b, a) ∈ Rel }
by simp

moreover from A2 have rel-respects-binary-pred { (a, b), (b, a) ∈ Rel } Pred
by blast

ultimately show ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ rel-respects-binary-pred Rel Pred
by blast
next
asssume ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ rel-respects-binary-pred Rel Pred
An encoding preserves, reflects, or respects a predicate iff indRelL reflects, preserves, or respects this predicate.

**Lemma (in encoding) enc-preserves-pred-iff-indRelL-reflects-pred:**

**Fixes** $\text{Pred} :: ('\text{procS}, '\text{procT}) \Rightarrow \text{bool}$

**Shows** $\text{enc-preserves-pred Pred = rel-reflects-pred indRelL Pred}$

**Using**

- $\text{enc-preserves-pred-iff-indRelL-reflects-pred[where Pred=Pred]}$
- $\text{indRelL-reflects-pred-iff-indRelL-reflects-pred[where Pred=Pred]}$

**By** blast

**Lemma (in encoding) enc-reflects-pred-iff-indRelL-preserves-pred:**

**Fixes** $\text{Pred} :: ('\text{procS}, '\text{procT}) \Rightarrow \text{bool}$

**Shows** $\text{enc-reflects-pred Pred = rel-preserves-pred indRelL Pred}$

**Using** $\text{enc-reflects-pred-iff-indRelL-preserves-pred[where Pred=Pred]}$

**By** blast

**Lemma (in encoding) enc-respects-pred-iff-indRelL-respects-pred:**

**Fixes** $\text{Pred} :: ('\text{procS}, '\text{procT}) \Rightarrow \text{bool}$

**Shows** $\text{enc-respects-pred Pred = rel-respects-pred indRelL Pred}$

**Using**

- $\text{enc-respects-pred-iff-indRelL-respects-pred[where Pred=Pred]}$
- $\text{indRelL-respects-pred-iff-indRelL-reflects-pred[where Pred=Pred]}$

**By** blast

An encoding preserves, reflects, or respects a predicate iff there exists a relation, namely indRelL, that relates literal translations with their source terms and reflects, preserves, or respects this predicate.

**Lemma (in encoding) enc-reflects-pred-iff-source-target-rel-reflects-pred:**

**Fixes** $\text{Pred} :: ('\text{procS}, '\text{procT}) \Rightarrow \text{bool}$

**Shows**

- $\text{enc-reflects-pred-iff-source-target-rel-reflects-pred[where Pred=Pred]}$
- $\text{indRelL-cond-reflection-iff-indRelL-cond-preservation[where Pred=Pred]}$

**By** simp

**Lemma (in encoding) enc-reflects-pred-iff-source-target-rel-preserved-pred:**

**Fixes** $\text{Pred} :: ('\text{procS}, '\text{procT}) \Rightarrow \text{bool}$

**Shows**

- $\text{enc-reflects-pred-iff-source-target-rel-preserved-pred[where Pred=Pred]}$
- $\text{indRelL-cond-preservation-iff-indRelL-cond-reflection[where Pred=Pred]}$

**By** simp

**Lemma (in encoding) enc-respects-pred-iff-source-target-rel-respects-pred:**

**Fixes** $\text{Pred} :: ('\text{procS}, '\text{procT}) \Rightarrow \text{bool}$

**Shows**

- $\text{enc-respects-pred-iff-source-target-rel-respects-pred[where Pred=Pred]}$
- $\text{indRelL-cond-respection-iff-indRelL-cond-respection[where Pred=Pred]}$

**By** simp
To analyse the respective of source term behaviours we use relations that contain both kind of pairs: (S, enc S) as well as (enc S, S) for all source terms S.

**inductive-set (in encoding) indRel**

```
:: (′procS, ′procT) Proc × (′procS, ′procT) Proc) set
where
encR: (SourceTerm S, TargetTerm ([S])) ∈ indRel |
encL: (TargetTerm ([S]), SourceTerm S) ∈ indRel
```

**abbreviation (in encoding) indRelInfix ::**

```
′procS, ′procT) Proc ⇒ (′procS, ′procT) Proc ⇒ bool (- R[]) - [75, 75] 80)
where
P R[] Q ≡ (P, Q) ∈ indRel
```

**lemma (in encoding) indRel-symm:**

```
shows sym indRel
unfolding sym-def
by (auto simp add: indRel.eq, simp indRel.eq, encR indRel.eq, encL)
```

**inductive-set (in encoding) indRelEQ**

```
:: (′procS, ′procT) Proc × (′procS, ′procT) Proc) set
where
encR: (SourceTerm S, TargetTerm ([S])) ∈ indRelEQ |
encL: (TargetTerm ([S]), SourceTerm S) ∈ indRelEQ |
target: (TargetTerm T, TargetTerm T) ∈ indRelEQ |
trans: [(P, Q) ∈ indRelEQ; (Q, R) ∈ indRelEQ] ⇒ (P, R) ∈ indRelEQ
```

**abbreviation (in encoding) indRelEQInfix ::**

```
′procS, ′procT) Proc ⇒ (′procS, ′procT) Proc ⇒ bool (- ~[]) - [75, 75] 80)
where
P ~[] Q ≡ (P, Q) ∈ indRelEQ
```

**lemma (in encoding) indRelEQ-refl:**

```
shows refl indRelEQ
unfolding refl-on-def
proof auto
fix P
show P ~[] P
proof (cases P)
case (SourceTerm SP)
  assume SP ∈ S P
  moreover have SourceTerm SP ~[] TargetTerm ([SP])
    by (rule indRelEQ.eq, encR)
  moreover have TargetTerm ([SP]) ~[] SourceTerm SP
    by (rule indRelEQ.eq, encL)
  ultimately show P ~[] P
    by (simp add: indRelEQ.trans[where P=SourceTerm SP and Q=TargetTerm ([SP])])
next
case (TargetTerm TP)
  assume TP ∈ T P
  thus P ~[] P
    by (simp add: indRelEQ.target)
qed
```

**lemma (in encoding) indRelEQ-is-preorder:**

```
shows preorder indRelEQ
unfolding preorder-on-def
proof
show refl indRelEQ
  by (rule indRelEQ-refl)
```
next

show trans indRelEQ
  unfolding trans-def
proof clarify
  fix P Q R
  assume P ~[\[] Q and Q ~[\[] R
  thus P ~[\[] R
    by (rule indRelEQ.trans)
qed
qed

lemma (in encoding) indRelEQ-symm:
  shows sym indRelEQ
  unfolding sym-def
proof clarify
  fix P Q
  assume P ~[\[] Q
  thus Q ~[\[] P
proof induct
  case (encR S)
    show TargetTerm ([S]) ~[\[] SourceTerm S
      by (rule indRelEQ.encL)
next
  case (encL S)
    show SourceTerm S ~[\[] TargetTerm ([S])
      by (rule indRelEQ.encR)
next
  case (target T)
    show TargetTerm T ~[\[] TargetTerm T
      by (rule indRelEQ.target)
next
  case (trans P Q R)
    assume R ~[\[] Q and Q ~[\[] P
    thus R ~[\[] P
      by (rule indRelEQ.trans)
qed
qed

lemma (in encoding) indRelEQ-is-equivalence:
  shows equivalence indRelEQ
  using indRelEQ-is-preorder indRelEQ-symm
  unfolding equiv-def preorder-on-def
by blast

lemma (in encoding) refl-trans-closure-of-indRel:
  shows indRelEQ = indRel*
proof auto
  fix P Q
  assume P ~[\[] Q
  thus (P, Q) \in indRel*
proof induct
  case (encR S)
    show (SourceTerm S, TargetTerm ([S])) \in indRel*
      using indRel.encR[of S]
      by simp
next
  case (encL S)
    show (TargetTerm ([S]), SourceTerm S) \in indRel*
      using indRel.encL[of S]
      by simp
next
case (target \( T \))
show (TargetTerm \( T \), TargetTerm \( T \)) \( \in \) indRel*
  by simp
next
  case (trans \( P \) \( Q \) \( R \))
  assume \( (P, Q) \in \) indRel* and \( (Q, R) \in \) indRel*
  thus \( (P, R) \in \) indRel*
    by simp
qed
next
  fix \( P \) \( Q \)
  assume \( (P, Q) \in \) indRel*
  thus \( P \sim \) \[ \cdot \] \( Q \)
    proof induct
      show \( P \sim \) \[ \cdot \] \( P \)
        using indRelEQ-refl
        unfolding refl-on-def
        by simp
    next
      case (step \( Q \) \( R \))
      assume \( P \sim \) \[ \cdot \] \( Q \)
      moreover assume \( Q \) \( R \) \[ \cdot \] \( R \)
      hence \( Q \sim \) \[ \cdot \] \( R \)
      by (induct, simp-all add: indRelEQ.encR indRelEQ.encL)
      ultimately show \( P \sim \) \[ \cdot \] \( R \)
        by (rule indRelEQ.trans)
    qed
  qed

lemma (in encoding) refl-symm-trans-closure-of-indRel:
  shows indRelEQ = (symcl (indRel=))^+
proof
  have (symcl (indRel=))^+ = (symcl indRel)^+
    by (rule refl-symm-trans-closure-is-symm-refl-trans-closure[where Rel=indRel])
  moreover have symcl indRel = indRel
    by (simp add: indRel-symm symm-closure-of-symm-rel[where Rel=indRel])
  ultimately show indRelEQ = (symcl (indRel=))^+
    by (simp add: refl-trans-closure-of-indRel)
  qed

lemma (in encoding) symm-closure-of-indRelR:
  shows indRel = symcl indRelR
  and indRelEQ = (symcl (indRelR=))^+
proof
  show indRel = symcl indRelR
    proof auto
      fix \( P \) \( Q \)
      assume \( P \) \( R \) \[ \cdot \] \( Q \)
      thus \( (P, Q) \in \) symcl indRelR
        by (induct, simp-all add: symcl-def indRelR.encR)
  next
    fix \( P \) \( Q \)
    assume \( (P, Q) \in \) symcl indRelR
    thus \( P \) \( R \) \[ \cdot \] \( Q \)
      by (auto simp add: symcl-def indRelR.encR.simps indRelR.encR indRelR.encL)
  qed
  thus indRelEQ = (symcl (indRelR=))^+
    using refl-symm-trans-closure-is-symm-refl-trans-closure[where Rel=indRelR]
    refl-trans-closure-of-indRel
    by simp
  qed
lemma (in encoding) symm-closure-of-indRelL:
  shows indRel = symcl indRelL
  and indRelEQ = (symcl (indRelL^=))^{+}
proof –
  show indRel = symcl indRelL
  proof auto
    fix P Q
    assume \( P \; R \; Q \)
    thus \((P, Q) \in symcl \; indRelL\) 
      by (induct, simp-all add: symcl-def indRelL_encL)
  next
    fix P Q
    assume \((P, Q) \in symcl \; indRelL\)
    thus \(P \; R \; Q\) 
      by (auto simp add: symcl-def indRelL_encL)
  qed
  thus \(\text{indRelEQ} = (symcl (\text{indRelL}^=))^{+}\)
  using refl-symm-trans-closure-is-symm-refl-trans-closure[where Rel=indRelL]
  by simp
qed

The relation \text{indRel} is a combination of \text{indRelL} and \text{indRelR}. \text{indRel} respects a predicate iff \text{indRelR} (or \text{indRelL}) respects it.

lemma (in encoding) indRel-respects-pred-iff-indRelR-respects-pred:
  fixes \text{Pred} :: (\'procS, \'procT) Proc \Rightarrow bool
  shows rel-respects-pred indRel \text{Pred} = rel-respects-pred indRelR \text{Pred}
proof
  assume \text{respection: rel-respects-pred indRel Pred}
  show rel-respects-pred indRelR \text{Pred}
  proof auto
    fix P Q
    assume \( P \; R \; Q \)
    from this obtain \( S \) where \( S \in S \; P \) and \( [S] \in T \; Q \) 
      by (induct, blast)
    hence \( P \; R \; Q \) 
      by (simp add: indRel_encR)
    moreover assume \text{Pred P}
    ultimately show \text{Pred Q}
      using respection
      by blast
  next
    fix P Q
    assume \( P \; R \; Q \)
    from this obtain \( S \) where \( S \in S \; P \) and \( [S] \in T \; Q \) 
      by (induct, blast)
    hence \( P \; R \; Q \) 
      by (simp add: indRel_encR)
    moreover assume \text{Pred Q}
    ultimately show \text{Pred P}
      using respection
      by blast
  qed
next
  assume rel-respects-pred indRelR \text{Pred}
  thus rel-respects-pred indRel \text{Pred}
    using symm-closure-of-indRelR(1)
    respection-and-closures(2)[where Rel=indRelR and Pred=Pred] 
  by blast
lemma (in encoding) \( \text{indRel-respects-binary-pred-iff-indRelR-respects-binary-pred} \):

fixes \( \text{Pred} \) :: \( (\text{procS}, \text{procT}) \rightarrow \text{Proc} \Rightarrow \text{\#} \rightarrow \text{bool} \)

shows \( \text{rel-respects-binary-pred \text{indRel Pred} = rel-respects-binary-pred \text{indRelR Pred} } \)

proof
assume \( \text{respection: rel-respects-binary-pred \text{indRel Pred} } \)
show \( \text{rel-respects-binary-pred \text{indRelR Pred} } \)
proof auto
fix \( P \) \( Q \) \( x \)
assume \( P \) \( R \) \( \lfloor S \rfloor \) \( R \) \( Q \)
from this obtain \( S \) where \( S \in S P \) and \( \lfloor S \rfloor \in T Q \)
by (induct, blast)

hence \( P \) \( R \) \( \lfloor S \rfloor \) \( Q \)
by (simp add: \text{indRel.encR})
moreover assume \( \text{Pred P x} \)
ultimately show \( \text{Pred Q x} \)
using \( \text{respection} \)
by blast
next
fix \( P \) \( Q \) \( x \)
assume \( P \) \( R \) \( \lfloor S \rfloor \) \( R \) \( Q \)
from this obtain \( S \) where \( S \in S P \) and \( \lfloor S \rfloor \in T Q \)
by (induct, blast)

hence \( P \) \( R \) \( \lfloor S \rfloor \) \( Q \)
by (simp add: \text{indRel.encR})
moreover assume \( \text{Pred Q x} \)
ultimately show \( \text{Pred P x} \)
using \( \text{respection} \)
by blast
qed

next
assume \( \text{rel-respects-binary-pred \text{indRelR Pred} } \)
thus \( \text{rel-respects-binary-pred \text{indRel Pred} } \)
using \text{symm-closure-of-indRelR}(1) \[ \text{respection-of-binary-predicates-and-closures}(2) \text{where Rel = indRelR and Pred = Pred} \]
by blast
qed

lemma (in encoding) \( \text{indRel-cond-respection-iff-indRelR-cond-respection} \):

fixes \( \text{Pred} \) :: \( (\text{procS}, \text{procT}) \rightarrow \text{Proc} \Rightarrow \text{\#} \rightarrow \text{bool} \)

shows \( \exists \text{Rel}. \ (orall S. (\text{SourceTerm S, TargetTerm } \lfloor S \rfloor) \in \text{Rel} \land (\text{TargetTerm } \lfloor S \rfloor, \text{SourceTerm S}) \in \text{Rel}) \land \text{rel-respects-pred Rel Pred} \)

proof
assume \( \exists \text{Rel}. \ (\forall S. (\text{SourceTerm S, TargetTerm } \lfloor S \rfloor) \in \text{Rel} \land (\text{TargetTerm } \lfloor S \rfloor, \text{SourceTerm S}) \in \text{Rel}) \land \text{rel-respects-pred Rel Pred} \)
from this obtain \( \text{Rel} \)
where \( \forall S. (\text{SourceTerm S, TargetTerm } \lfloor S \rfloor) \in \text{Rel} \land (\text{TargetTerm } \lfloor S \rfloor, \text{SourceTerm S}) \in \text{Rel} \land \text{rel-respects-pred Rel Pred} \)
by blast
thus \( \exists \text{Rel}. \ (\forall S. (\text{SourceTerm S, TargetTerm } \lfloor S \rfloor) \in \text{Rel} \land \text{rel-respects-pred Rel Pred} \)
by blast
next
assume \( \exists \text{Rel}. \ (\forall S. (\text{SourceTerm S, TargetTerm } \lfloor S \rfloor) \in \text{Rel} \land \text{rel-respects-pred Rel Pred} \)
from this obtain \( \text{Rel} \) where \( A1 : \forall S. (\text{SourceTerm S, TargetTerm } \lfloor S \rfloor) \in \text{Rel} \land \text{rel-respects-pred Rel Pred} \)
and \( A2 : \text{rel-respects-pred Rel Pred} \)
by blast
from \( A1 \) have \( \forall S. (\text{SourceTerm S, TargetTerm } \lfloor S \rfloor) \in \text{symcl Rel} \land (\text{TargetTerm } \lfloor S \rfloor, \text{SourceTerm S}) \in \text{symcl Rel} \)
by (simp add: symcl-def)
moreover from A2 have rel-respects-pred (symcl Rel) Pred
  using respection-and-closures(2)[where Rel=Rel and Pred=Pred]
by blast
ultimately
show \exists\, Rel. (\forall\, S. (SourceTerm S, TargetTerm (\[ [S] \])) \in Rel \land (TargetTerm (\[ [S] \)), SourceTerm S) \in Rel)
  \land rel-respects-pred Rel Pred
by blast
qed

lemma (in encoding) indRel-cond-binary-respection-iff-indRelR-cond-binary-respection:
fixes Pred :: ('procS, 'procT) Proc \Rightarrow 'b \Rightarrow bool
shows (\exists\, Rel.
  (\forall\, S. (SourceTerm S, TargetTerm (\[ [S] \])) \in Rel \land (TargetTerm (\[ [S] \)), SourceTerm S) \in Rel)
  \land rel-respects-binary-pred Rel Pred
= (\exists\, Rel. (\forall\, S. (SourceTerm S, TargetTerm (\[ [S] \])) \in Rel)
  \land rel-respects-binary-pred Rel Pred)
proof
assume \exists\, Rel. (\forall\, S. (SourceTerm S, TargetTerm (\[ [S] \])) \in Rel
  \land (TargetTerm (\[ [S] \)), SourceTerm S) \in Rel) \land rel-respects-binary-pred Rel Pred
from this obtain Rel
  where \forall\, S. (SourceTerm S, TargetTerm (\[ [S] \])) \in Rel \land (TargetTerm (\[ [S] \)), SourceTerm S) \in Rel
  \land rel-respects-binary-pred Rel Pred
by blast
thus \exists\, Rel. (\forall\, S. (SourceTerm S, TargetTerm (\[ [S] \])) \in Rel) \land rel-respects-binary-pred Rel Pred
by blast
next
assume \exists\, Rel. (\forall\, S. (SourceTerm S, TargetTerm (\[ [S] \])) \in Rel) \land rel-respects-binary-pred Rel Pred
from this obtain Rel where A1: \forall\, S. (SourceTerm S, TargetTerm (\[ [S] \])) \in Rel
  \land A2: rel-respects-binary-pred Rel Pred
by blast
from A1 have \forall\, S. (SourceTerm S, TargetTerm (\[ [S] \])) \in symcl Rel
  \land (TargetTerm (\[ [S] \)), SourceTerm S) \in symcl Rel
by (simp add: symcl-def)
moreover from A2 have rel-respects-binary-pred (symcl Rel) Pred
  using respection-of-binary-predicates-and-closures(2)[where Rel=Rel and Pred=Pred]
by blast
ultimately
show \exists\, Rel. (\forall\, S. (SourceTerm S, TargetTerm (\[ [S] \])) \in Rel \land (TargetTerm (\[ [S] \)), SourceTerm S) \in Rel)
  \land rel-respects-binary-pred Rel Pred
by blast
qed

An encoding respects a predicate iff \( indRel \) respects this predicate.

lemma (in encoding) enc-respects-pred-iff-indRel-respects-pred:
fixes Pred :: ('procS, 'procT) Proc \Rightarrow bool
shows enc-respects-pred Pred = rel-respects-pred indRel Pred
  using enc-respects-pred-iff-indRelR-respects-pred[where Pred=Pred]
  indRel-respects-pred-iff-indRelR-respects-pred[where Pred=Pred]
  by simp

An encoding respects a predicate iff there exists a relation, namely \( indRel \), that relates source terms and their literal translations in both directions and respects this predicate.

lemma (in encoding) enc-respects-pred-iff-source-target-rel-respects-pred-encRL:
fixes Pred :: ('procS, 'procT) Proc \Rightarrow bool
shows enc-respects-pred Pred
  = (\exists\, Rel.
    (\forall\, S. (SourceTerm S, TargetTerm (\[ [S] \])) \in Rel \land (TargetTerm (\[ [S] \)), SourceTerm S) \in Rel)
    \land rel-respects-pred Rel Pred)
  using enc-respects-pred-iff-source-target-rel-respects-pred-encRL[where Pred=Pred]

76
5.2 Relations Induced by the Encoding and a Relation on Target Terms

Some encodability like e.g. operational correspondence are defined w.r.t. a relation on target terms. To analyse such criteria we include the respective target term relation in the considered relation on the disjoint union of source and target terms.

**inductive-set (in encoding) indRelRT**

```
for TRel :: ('procT × 'procT) set
where
encR: (SourceTerm S, TargetTerm ([S])) ∈ indRelRT TRel |
target: (T1, T2) ∈ TRel ⇒ (TargetTerm T1, TargetTerm T2) ∈ indRelRT TRel
```

**abbreviation (in encoding) indRelRTinfix**

```
:: ('procS, 'procT) Proc ⇒ ('procT × 'procT) set ⇒ ('procS, 'procT) Proc ⇒ bool
(- R[-]RT<> - [75, 75, 75] 80)
where
P R[-]RT<TRel> Q ≡ (P, Q) ∈ indRelRT TRel
```

**inductive-set (in encoding) indRelRTPO**

```
:: ('procT × 'procT) set ⇒ (((('procS, 'procT) Proc) × (('procS, 'procT) Proc)) set
for TRel :: ('procT × 'procT) set
where
encR: (SourceTerm S, TargetTerm ([S])) ∈ indRelRTPO TRel |
source: (SourceTerm S, SourceTerm S) ∈ indRelRTPO TRel |
target: (T1, T2) ∈ TRel ⇒ (TargetTerm T1, TargetTerm T2) ∈ indRelRTPO TRel |
trans: [(P, Q) ∈ indRelRTPO TRel; (Q, R) ∈ indRelRTPO TRel] ⇒ (P, R) ∈ indRelRTPO TRel
```

**abbreviation (in encoding) indRelRTPOinfix**

```
:: ('procS, 'procT) Proc ⇒ ('procT × 'procT) set ⇒ ('procS, 'procT) Proc ⇒ bool
(- ≲[-]RT<-> - [75, 75, 75] 80)
where
P ≲[-]RT<TRel> Q ≡ (P, Q) ∈ indRelRTPO TRel
```

**lemma (in encoding) indRelRTPO-refl:**

```
fixes TRel :: ('procT × 'procT) set
assumes refl: refl TRel
shows refl (indRelRTPO TRel)
unfolding refl-on-def

proof auto
fix P
show P ≲[-]RT<TRel> P
proof (cases P)
  case (SourceTerm SP)
  assume SP ∈ S P
  thus P ≲[-]RT<TRel> P
    by (simp add: indRelRTPO.source)
next
  case (TargetTerm TP)
  assume TP ∈ T P
  with refl show P ≲[-]RT<TRel> P
    unfolding refl-on-def
    by (simp add: indRelRTPO.target)
qed

```

**lemma (in encoding) refl-trans-closure-of-indRelRT:**

```
fixes TRel :: ('procT × 'procT) set
```
assumes refl: refl TRel
shows indRelRTPO TRel = (indRelRT TRel)^*
proof auto
fix P Q
assume P ≤[\]RT<TRel> Q
thus (P, Q) ∈ (indRelRT TRel)^*
proof induct
  case (encR S)
  show (SourceTerm S, TargetTerm ([S])) ∈ (indRelRT TRel)^*
    using indRelRT.encR[of S TRel]
    by simp
next
case (source S)
show (SourceTerm S, SourceTerm S) ∈ (indRelRT TRel)^*
  by simp
next
case (target T1 T2)
assume (T1, T2) ∈ TRel
thus (TargetTerm T1, TargetTerm T2) ∈ (indRelRT TRel)^*
  using indRelRT.target[of T1 T2 TRel]
  by simp
next
case (trans P Q R)
assume (P, Q) ∈ (indRelRT TRel)^* and (Q, R) ∈ (indRelRT TRel)^*
thus (P, R) ∈ (indRelRT TRel)^*
  by simp
qed
next
fix P Q
assume (P, Q) ∈ (indRelRT TRel)^*
thus P ≤[\]RT<TRel> Q
proof induct
from refl show P ≤[\]RT<TRel> P
  unfolding refl-on-def
  by simp
next
case (step Q R)
assume P ≤[\]RT<TRel> Q
moreover assume Q ≤[\]RT<TRel> R
hence Q ≤[\]RT<TRel> R
  by (induct, simp-all add: indRelRTPO.encR indRelRTPO.target)
ultimately show P ≤[\]RT<TRel> R
  by (rule indRelRTPO.trans)
qed
qed

lemma (in encoding) indRelRTPO-is-preorder:
  fixes TRel :: ('procT × 'procT) set
  assumes reflT: refl TRel
  shows preorder (indRelRTPO TRel)
  unfolding preorder-on-def
proof
  from reflT show refl (indRelRTPO TRel)
    by (rule indRelRTPO-refl)
next
  show trans (indRelRTPO TRel)
    unfolding trans-def
    by clarify
  fix P Q R
  assume P ≤[\]RT<TRel> Q and Q ≤[\]RT<TRel> R
Thus $P \leq \llbracket RT < TRel \rrbracket R$

using $\text{indRelRTPO.trans}$

by blast

qed

\text{lemma (in encoding) transitive-closure-of-$TRel$-to-$indRelRTPO$:}

fixes $TRel :: ('procT \times 'procT) \Rightarrow bool$

and $TP TQ :: 'procT$

shows $(TP, TQ) \in TRel^+ \implies \text{TargetTerm}$ $TP \leq \llbracket RT < TRel \rrbracket$ $\text{TargetTerm}$ $TQ$

proof

assume $(TP, TQ) \in TRel$

thus $\text{TargetTerm}$ $TP \leq \llbracket RT < TRel \rrbracket$ $\text{TargetTerm}$ $TQ$

by (rule $\text{indRelRTPO.target}$)

next

\text{case (step $TQ TR$)}

assume $\text{TargetTerm}$ $TP \leq \llbracket RT < TRel \rrbracket$ $\text{TargetTerm}$ $TQ$

moreover assume $(TQ, TR) \in TRel$

hence $\text{TargetTerm}$ $TP \leq \llbracket RT < TRel \rrbracket$ $\text{TargetTerm}$ $TR$

by (simp add: $\text{indRelRTPO.target}$)

ultimately show $\text{TargetTerm}$ $TP \leq \llbracket RT < TRel \rrbracket$ $\text{TargetTerm}$ $TR$

by (rule $\text{indRelRTPO.trans}$)

qed

\text{qed}

The relation $\text{indRelRT}$ is the smallest relation that relates all source terms and their literal translations and contains $TRel$. Thus there exists a relation that relates source terms and their literal translations and satisfies some predicate on its pairs iff the predicate holds for the pairs of $\text{indRelR}$.

\text{lemma (in encoding) $\text{indRelR}$-modulo-pred-impl-$\text{indRelRT}$-modulo-pred:}

fixes $Pred :: ('procS \times 'procT) \Rightarrow bool$

shows $\forall (P, Q) \in \text{indRelR. Pred (P, Q)} = (\forall TRel. (\forall (TP, TQ) \in TRel. Pred (TargetTerm TP, TargetTerm TQ)) 

= (\forall (P, Q) \in \text{indRelRT TRel. Pred (P, Q)})$

proof (rule \text{iff})

assume $A: \forall (P, Q) \in \text{indRelR. Pred (P, Q)}$

show $\forall TRel. (\forall (TP, TQ) \in TRel. Pred (TargetTerm TP, TargetTerm TQ)) 

= (\forall (P, Q) \in \text{indRelRT TRel. Pred (P, Q)})$

proof (rule \text{allI, rule \text{iff}})

fix $TRel$

assume $\forall (TP, TQ) \in TRel. Pred (TargetTerm TP, TargetTerm TQ)$

with $A$ show $\forall (P, Q) \in \text{indRelRT TRel. Pred (P, Q)}$

by (auto simp add: $\text{indRelR.encR}$ $\text{indRelRT.simps}$)

next

fix $TRel$

assume $\forall (P, Q) \in \text{indRelRT TRel. Pred (P, Q)}$

thus $\forall (TP, TQ) \in TRel. Pred (TargetTerm TP, TargetTerm TQ)$

by (auto simp add: $\text{indRelRT.target}$)

qed

next

assume $\forall TRel. (\forall (TP, TQ) \in TRel. Pred (TargetTerm TP, TargetTerm TQ)) 

= (\forall (P, Q) \in \text{indRelRT TRel. Pred (P, Q)})$

hence $B: \forall TRel. (\forall (TP, TQ) \in TRel. Pred (TargetTerm TP, TargetTerm TQ)) 

= (\forall (P, Q) \in \text{indRelRT TRel. Pred (P, Q)})$

by blast

have $A: \text{Pred (SourceTerm S, TargetTerm (\llbracket S \rrbracket))}$

using $B[\{\}]

by (simp add: $\text{indRelRT.simps}$)

79
thus ∀(P, Q) ∈ indRelR. Pred (P, Q) 
by (auto simp add: indRelR.simps)

qed

lemma (in encoding) indRelRT-iff-exists-source-target-relation:
  fixes Pred :: ('procS, 'procT) Proc × ('procS, 'procT) Proc ⇒ bool
  shows (∀ TRel. (∀ (TP, TQ) ∈ TRel. Pred (TargetTerm TP, TargetTerm TQ))
  ¬→ (∀ (P, Q) ∈ indRelRT TRel. Pred (P, Q))))
  = (∃ Rel. (∃ S. (SourceTerm S, TargetTerm ([(S)]) ∈ Rel) ∧ (∀ (P, Q) ∈ Rel. Pred (P, Q))))
  using indRelRT-iff-exists-source-target-relation[where Pred=Pred]
  indRelR-modulo-pred-impl-indRelRT-modulo-pred[where Pred=Pred]
  by simp

lemma (in encoding) indRelRT-modulo-pred-impl-indRelRTPO-modulo-pred:
  fixes TRel :: ('procT × 'procT) set
  and Pred :: ('procS, 'procT) Proc × ('procS, 'procT) Proc ⇒ bool
  assumes reflCond: ∀ P. Pred (P, P)
  and transCond: ∀ P Q R. Pred (P, Q) ∧ Pred (Q, R) ⇒ Pred (P, R)
  shows (∀ (P, Q) ∈ indRelRT TRel. Pred (P, Q)) = (∀ (P, Q) ∈ indRelRTPO TRel. Pred (P, Q))
proof auto
  fix P Q
  assume A: ∀ x ∈ indRelRT TRel. Pred x
  assume P ≤⇧TRel< TRel> Q
  thus Pred (P, Q)
proof induct
  case (encR S)
  have SourceTerm S R[plan TRel< TRel> TargetTerm ([(S)])
    by (simp add: indRelRT.encR)
  with A show Pred (SourceTerm S, TargetTerm ([(S)])
    by simp
next
  case (source S)
  from reflCond show Pred (SourceTerm S, SourceTerm S)
    by simp
next
  case (target T1 T2)
  assume (T1, T2) ∈ TRel
  hence TargetTerm T1 R[plan TRel< TRel> TargetTerm T2
    by (simp add: indRelRT.target)
  with A show Pred (TargetTerm T1, TargetTerm T2)
    by simp
next
  case (trans P Q R)
  assume Pred (P, Q) and Pred (Q, R)
  with transCond show Pred (P, R)
    by blast
qed

lemma (in encoding) indRelR-modulo-pred-impl-indRelRTPO-modulo-pred:
  fixes Pred :: ('procS, 'procT) Proc × ('procS, 'procT) Proc ⇒ bool
  assumes ∀ P. Pred (P, P)
  and ∀ P Q R. Pred (P, Q) ∧ Pred (Q, R) ⇒ Pred (P, R)
  shows (∀ (P, Q) ∈ indRelR. Pred (P, Q))
  = (∀ TRel. (∀ (TP, TQ) ∈ TRel. Pred (TargetTerm TP, TargetTerm TQ))
  ¬→ (∀ (P, Q) ∈ indRelRTPO TRel. Pred (P, Q))))

80
proof –

have \((\forall (P, Q) \in \text{indRelR}) \cdot \text{Pred} (P, Q) = (\forall T \text{Rel.} (\forall (TP, TQ) \in T \text{Rel.} \cdot (\forall (P, Q) \in \text{indRelRT} \cdot T \text{Rel.} \cdot \text{Pred} (P, Q)))\))

using \text{indRelR-modulo-pred-impl-indRelRT-modulo-pred}[\text{where} \text{Pred} = \text{Pred}] by simp

moreover

have \(\forall T \text{Rel.} (\forall (P, Q) \in \text{indRelRT} \cdot T \text{Rel.} \cdot \text{Pred} (P, Q)) = (\forall (P, Q) \in \text{indRelRTPO} \cdot T \text{Rel.} \cdot \text{Pred} (P, Q))\)

using \text{assms indRelRT-modulo-pred-impl-indRelRTPO-modulo-pred}[\text{where} \text{Pred} = \text{Pred}] by blast

ultimately show \(?\text{thesis}\)

by simp

qed

The relation \text{indRelLT} includes \text{TRel} and relates literal translations and their source terms.

\textbf{inductive-set (in encoding) indRelLT}

:\((\text{procT} \times \text{procT})\) set \(\Rightarrow\) \(((\text{procS}, \text{procT})\) \text{Proc} \times ((\text{procS}, \text{procT})\) \text{Proc} \Rightarrow \text{bool}

\((\cdot \text{encL} - \cdot \text{LT} < - \cdot \text{75}, \text{75}, \text{75} \cdot \text{80})\)

where

\(\text{Pred} \cdot \text{LT} < T\text{Rel} > \text{Q} \equiv (P, Q) \in \text{indRelLT} \cdot T\text{Rel}\)

\textbf{inductive-set (in encoding) indRelLTinfix}

:\((\text{procS}, \text{procT})\) \text{Proc} \Rightarrow ((\text{procT} \times \text{procT})\) set \(\Rightarrow\) \(((\text{procS}, \text{procT})\) \text{Proc} \Rightarrow \text{bool}

\((\cdot \text{encL} - \cdot \text{LT} < - \cdot \text{75}, \text{75}, \text{75} \cdot \text{80})\)

where

\(\text{Pred} \cdot \text{LT} < T\text{Rel} > Q \equiv (P, Q) \in \text{indRelLT} \cdot T\text{Rel}\)

\textbf{inductive-set (in encoding) indRelLTPOinfix}

:\((\text{procS}, \text{procT})\) \text{Proc} \Rightarrow ((\text{procT} \times \text{procT})\) set \(\Rightarrow\) \(((\text{procS}, \text{procT})\) \text{Proc} \Rightarrow \text{bool}

\((\cdot \text{encL} - \cdot \text{LT} < - \cdot \text{75}, \text{75}, \text{75} \cdot \text{80})\)

where

\(\text{Pred} \cdot \text{LT} < T\text{Rel} > Q \equiv (P, Q) \in \text{indRelLTPO} \cdot T\text{Rel}\)

\textbf{abbreviation (in encoding) indRelLTPO-refl:}

\textbf{fixes} \(T\text{Rel} \vdash (\text{procT} \times \text{procT})\) set

\textbf{assumes} refl: refl \(T\text{Rel}\)

\textbf{shows} refl (\text{indRelLTPO} \cdot T\text{Rel})

\textbf{unfolding} refl-on-def

\textbf{proof auto}

\textbf{fix} \(P\)

\textbf{show} \(P \leq \cdot \text{LT} < T\text{Rel} > P\)

\textbf{proof (cases} \(P\))

\textbf{case} \((\text{SourceTerm} SP)\)

\textbf{assume} \(SP \in S \cdot P\)

\textbf{thus} \(P \leq \cdot \text{LT} < T\text{Rel} > P\)

\textbf{by (simp add: indRelLTPO.source)}

\textbf{next}

\textbf{case} \((\text{TargetTerm} TP)\)

\textbf{assume} \(TP \in T \cdot P\)

\textbf{with} \text{refl} \textbf{show} \(P \leq \cdot \text{LT} < T\text{Rel} > P\)

\textbf{using} \text{indRelLTPO.target[of} \(TP \cdot TP \cdot T\text{Rel}])

\textbf{unfolding} refl-on-def
lemma (in encoding) refl-trans-closure-of-indRelLT:
  fixes TRel :: ('procT × 'procT) set
  assumes refl: refl TRel
  shows indRelLTPO TRel = (indRelLT TRel)∗
proof auto
  fix P Q
  assume P ≲· LT < TRel > Q
  thus (P, Q) ∈ (indRelLT TRel)∗
proof induct
  case (encL S)
  show (TargetTerm ([S]), SourceTerm S) ∈ (indRelLT TRel)∗
    using indRelLT_encL[of S TRel]
      by simp
next
  case (source S)
  show (SourceTerm S, SourceTerm S) ∈ (indRelLT TRel)∗
    by simp
next
  case (target T1 T2)
  assume (T1, T2) ∈ TRel
  thus (TargetTerm T1, TargetTerm T2) ∈ (indRelLT TRel)∗
    using indRelLT_target[of T1 T2 TRel]
      by simp
next
  case (trans P Q R)
  assume (P, Q) ∈ (indRelLT TRel)∗ and (Q, R) ∈ (indRelLT TRel)∗
  thus (P, R) ∈ (indRelLT TRel)∗
    by simp
next
  fix P Q
  assume (P, Q) ∈ (indRelLT TRel)∗
  thus P ≲· LT < TRel > Q
proof induct
  from refl show P ≲· LT < TRel > P
    using indRelLTPO_refl[of TRel]
      unfolding refl-on-def
      by simp
next
  case (step Q R)
  assume P ≲· LT < TRel > Q
  moreover assume Q R ≲· LT < TRel > R
  hence Q ≲· LT < TRel > R
    by (induct, simp-all add: indRelLTPO_encL indRelLTPO_target)
  ultimately show P ≲· LT < TRel > R
    by (rule indRelLTPO_trans)
  qed
next

inductive-set (in encoding) indRelT
  :: ('procT × 'procT) set ⇒ (((procS, 'procT) Proc) × ((procS, 'procT) Proc)) set
  for TRel :: ('procT × 'procT) set
where
  encR: (SourceTerm S, TargetTerm ([S])) ∈ indRelT TRel
  | encL: (TargetTerm ([S]), SourceTerm S) ∈ indRelT TRel
  | target: (T1, T2) ∈ TRel ⇒ (TargetTerm T1, TargetTerm T2) ∈ indRelT TRel
abbreviation (in encoding) indRelTinfix
:: (procS, procT) Proc ⇒ (procT × procT) Proc ⇒ bool
(- R[\cdot] T<-> - [75, 75, 75] 80)

where
P R[\cdot] T<TRel> Q ≡ (P, Q) ∈ indRelT TRel

lemma (in encoding) indRelT-symm:
fixes TRel :: (procT × procT) set
assumes symm: sym TRel
shows sym (indRelT TRel)
unfolding sym-def

proof clarify
fix P Q
assume (P, Q) ∈ indRelT TRel
thus (Q, P) ∈ indRelT TRel
using symm
unfolding sym-def
by (induct, simp-all add: indRelT.encL indRelT.encR indRelT.target)

qed

inductive-set (in encoding) indRelTEQ
:: (procT × procT) set ⇒ ((procS, procT) Proc) × ((procS, procT) Proc) set
for TRel :: (procT × procT) set
where
encR: (SourceTerm S, TargetTerm (\{S\})) ∈ indRelTEQ TRel |
encL: (TargetTerm (\{S\}), SourceTerm S) ∈ indRelTEQ TRel |
target: (T1, T2) ∈ TRel ⇒ (TargetTerm T1, TargetTerm T2) ∈ indRelTEQ TRel |
trans: [(P, Q) ∈ indRelTEQ TRel; (Q, R) ∈ indRelTEQ TRel] ⇒ (P, R) ∈ indRelTEQ TRel

abbreviation (in encoding) indRelTEQinfix
:: (procS, procT) Proc ⇒ (procT × procT) Proc ⇒ bool
(- ~[\cdot] T<-> - [75, 75, 75] 80)

where
P ~[\cdot] T<TRel> Q ≡ (P, Q) ∈ indRelTEQ TRel

lemma (in encoding) indRelTEQ-refl:
fixes TRel :: (procT × procT) set
assumes refl: refl TRel
shows refl (indRelTEQ TRel)
unfolding refl-on-def

proof auto
fix P
show P ~[\cdot] T<TRel> P
proof (cases P)
case (SourceTerm SP)
assume SP ∈ S P
moreover have SourceTerm SP ~[\cdot] T<TRel> TargetTerm (\{SP\})
by (rule indRelTEQ.encR)
moreover have TargetTerm (\{SP\}) ~[\cdot] T<TRel> SourceTerm SP
by (rule indRelTEQ.encL)
ultimately show P ~[\cdot] T<TRel> P
by (simp add: indRelTEQ.trans[where P=SourceTerm SP and Q=TargetTerm (\{SP\})])
next
case (TargetTerm TP)
assume TP ∈ T P
with refl show P ~[\cdot] T<TRel> P
unfolding refl-on-def
by (simp add: indRelTEQ.target)

qed

qed

83
lemma (in encoding) indRelTEQ-symm:
  fixes TRel :: (′procT × ′procT) set
  assumes symm: sym TRel
  shows sym (indRelTEQ TRel)
  unfolding sym-def
proof clarify
  fix P Q
  assume P ∼\[\cdot\] T<TRel> Q
  thus Q ∼\[\cdot\] T<TRel> P
proof induct
  case (encR S)
  show TargetTerm ([S]) ∼\[\cdot\] T<TRel> SourceTerm S
    by (rule indRelTEQ.encR)
next
  case (encL S)
  show SourceTerm S ∼\[\cdot\] T<TRel> TargetTerm ([S])
    by (rule indRelTEQ.encL)
next
  case (target T1 T2)
  assume (T1, T2) ∈ TRel
  with symm show TargetTerm T2 ∼\[\cdot\] T<TRel> TargetTerm T1
    unfolding sym-def
    by (simp add: indRelTEQ.target)
next
  case (trans P Q R)
  assume R ∼\[\cdot\] T<TRel> Q and Q ∼\[\cdot\] T<TRel> P
  thus R ∼\[\cdot\] T<TRel> P
    by (rule indRelTEQ.trans)
qed

lemma (in encoding) refl-trans-closure-of-indRelT:
  fixes TRel :: (′procT × ′procT) set
  assumes refl: refl TRel
  shows indRelTEQ TRel = (indRelT TRel)*
proof auto
  fix P Q
  assume P ∼\[\cdot\] T<TRel> Q
  thus (P, Q) ∈ (indRelT TRel)*
proof induct
  case (encR S)
  show (SourceTerm S, TargetTerm ([S])) ∈ (indRelT TRel)*
    using indRelT.encR[of S TRel]
    by simp
next
  case (encL S)
  show (TargetTerm ([S]), SourceTerm S) ∈ (indRelT TRel)*
    using indRelT.encL[of S TRel]
    by simp
next
  case (target T1 T2)
  assume (T1, T2) ∈ TRel
  thus (TargetTerm T1, TargetTerm T2) ∈ (indRelT TRel)*
    using indRelT.target[of T1 T2 TRel]
    by simp
next
  case (trans P Q R)
  assume (P, Q) ∈ (indRelT TRel)* and (Q, R) ∈ (indRelT TRel)*
  thus (P, R) ∈ (indRelT TRel)*
    by simp
qed
next
  fix \( P, Q \)
  assume \((P, Q) \in (\text{indRelT \ TRel})^*\)
  thus \( P \sim [\cdot]T<\text{TRel}> Q \)
proof induct
  from refl show \( P \sim [\cdot]T<\text{TRel}> P \)
    using indRelTEQ-refl[of \text{TRel}]
  unfolding refl-on-def
    by simp
next
  case (step \( Q, R \))
    assume \( P \sim [\cdot]T<\text{TRel}> Q \)
  moreover assume \( Q \sim [\cdot]T<\text{TRel}> R \)
  hence \( Q \sim [\cdot]T<\text{TRel}> R \)
    by (induct, simp-all add: indRelTEQ.encR indRelTEQ.encL indRelTEQ.target)
  ultimately show \( P \sim [\cdot]T<\text{TRel}> R \)
    by (rule indRelTEQ.trans)
  qed

lemma (in encoding) refl-symm-trans-closure-of-indRelT:
  fixes \( \text{TRel} :: (\text{procT} \times \text{procT}) \text{ set} \)
  assumes refl: refl \text{TRel}
    and symm: sym \text{TRel}
  shows \( \text{indRelTEQ} \ TRel = (\text{symcl} ((\text{indRelT} \ TRel)^=))^+ \)
proof -
  have \( (\text{symcl} ((\text{indRelT} \ TRel)^=))^+ = (\text{symcl} ((\text{indRelT} \ TRel))^*) \)
    by (rule refl-symm-trans-closure-is-symm-refl-trans-closure[where \text{Rel}=\text{indRelT} \ TRel])
  moreover from symm have \( \text{symcl} ((\text{indRelT} \ TRel) = \text{symcl} \text{TRel}) \)
    using indRelT-symm[where \text{TRel}=\text{TRel}] symm-closure-of-symm-rel[where \text{Rel}=\text{indRelT} \ TRel]
    by blast
  ultimately show \( \text{indRelTEQ} \ TRel = (\text{symcl} ((\text{indRelT} \ TRel)^=))^+ \)
    using refl refl-trans-closure-of-indRelT[where \text{TRel}=\text{TRel}]
    by simp
  qed

lemma (in encoding) symm-closure-of-indRelRT:
  fixes \( \text{TRel} :: (\text{procT} \times \text{procT}) \text{ set} \)
  assumes refl: refl \text{TRel}
    and symm: sym \text{TRel}
  shows \( \text{indRelT} \ TRel = \text{symcl} \ (\text{indRelRT} \ TRel) \)
    and \( \text{indRelTEQ} \ TRel = (\text{symcl} ((\text{indRelRT} \ TRel)^=))^+ \)
proof auto
  show \( \text{indRelT} \ TRel = \text{symcl} \ (\text{indRelRT} \ TRel) \)
  proof
    fix \( P \)
    assume \( P \sim [\cdot]T<\text{TRel}> Q \)
    thus \( (P, Q) \in \text{symcl} \ (\text{indRelRT} \ TRel) \)
      by (induct, simp-all add: symcl-def indRelRT.encR indRelRT.target)
  next
    fix \( P \)
    assume \( (P, Q) \in \text{symcl} \ (\text{indRelRT} \ TRel) \)
    thus \( P \sim [\cdot]T<\text{TRel}> Q \)
  proof (auto simp add: symcl-def indRelRT.simps)
    fix \( S \)
    show \( \text{SourceTerm} \ S \sim [\cdot]T<\text{TRel}> \text{TargetTerm} ([S]) \)
      by (rule indRelT.encR)
  next
    fix \( T1, T2 \)
    assume \( (T1, T2) \in \text{TRel} \)
    thus \( \text{TargetTerm} \ T1 \sim [\cdot]T<\text{TRel}> \text{TargetTerm} \ T2 \)
by (rule indRelT.target)
next
fix S
show TargetTerm ([S]) \text{R} \llbracket \ell \rrbracket T < \text{TRel} > \text{SourceTerm S}
by (rule indRelT.encL)
next
fix T1 T2
assume (T1, T2) \in \text{TRel}
with symm show TargetTerm T2 \text{R} \llbracket \ell \rrbracket T < \text{TRel} > \text{TargetTerm T1}
  unfolding sym-def
by (simp add: indRelT.target)
qed
qed
with refl show \text{indRelTEQ} \text{TRel} = (\text{symcl} ((\text{indRelRT} \text{TRel})\gamma))^{\ast}
using refl-symm-trans-closure-is-symm-refl-trans-closure[where Rel=\text{indRelRT} \text{TRel}]
refl-trans-closure-of-indRelT
by simp

lemma (in encoding) symm-closure-of-indRelLT:
fixes \text{TRel} :: ('procT \times 'procT) set
assumes refl: refl \text{TRel}
and symm: sym \text{TRel}
shows \text{indRelLT} \text{TRel} = \text{symcl} (\text{indRelLT} \text{TRel})
and \text{indRelTEQ} \text{TRel} = (\text{symcl} ((\text{indRelLT} \text{TRel})\gamma))^{\ast}
proof --
show \text{indRelLT} \text{TRel} = \text{symcl} (\text{indRelLT} \text{TRel})
proof auto
fix P Q
assume \((P, Q) \in \text{symcl} (\text{indRelLT} \text{TRel})\)
thus \((P, Q) \in \text{symcl} (\text{indRelLT} \text{TRel})\)
  by (induct, simp-all add: symcl-def indRelLT.encL indRelLT.target)
next
fix P Q
assume \((P, Q) \in \text{symcl} (\text{indRelLT} \text{TRel})\)
thus \((P, Q) \in \text{symcl} (\text{indRelLT} \text{TRel})\)
proof (auto simp add: symcl-def indRelLT.simps)
fix S
show SourceTerm S \text{R} \llbracket \ell \rrbracket T < \text{TRel} > \text{TargetTerm ([S])}
  by (rule indRelT.encR)
next
fix T1 T2
assume (T1, T2) \in \text{TRel}
thus TargetTerm T1 \text{R} \llbracket \ell \rrbracket T < \text{TRel} > TargetTerm T2
  by (rule indRelT.target)
next
fix S
show TargetTerm ([S]) \text{R} \llbracket \ell \rrbracket T < \text{TRel} > \text{SourceTerm S}
  by (rule indRelT.encL)
next
fix T1 T2
assume (T1, T2) \in \text{TRel}
with symm show TargetTerm T2 \text{R} \llbracket \ell \rrbracket T < \text{TRel} > \text{TargetTerm T1}
  unfolding sym-def
by (simp add: indRelT.target)
qed
qed
with refl show \text{indRelTEQ} \text{TRel} = (\text{symcl} ((\text{indRelLT} \text{TRel})\gamma))^{\ast}
using refl-symm-trans-closure-is-symm-refl-trans-closure[where Rel=\text{indRelLT} \text{TRel}]
refl-trans-closure-of-indRelT
by simp
If the relations indRelRT, indRelLT, or indRelT contain a pair of target terms, then this pair is also related by the considered target term relation.

**Lemma (in encoding) indRelRT-to-TRel:**

- **Fixes** TRel :: (procT × procT) set
- **And** TP TQ :: 'procT
- **Assumes** rel: TargetTerm TP R[\[RT\]<TRel>] TargetTerm TQ
- **Shows** (TP, TQ) ∈ TRel
  - **Using** rel
  - by (simp add: indRelRT.simps)

**Lemma (in encoding) indRelLT-to-TRel:**

- **Fixes** TRel :: (procT × procT) set
- **And** TP TQ :: 'procT
- **Assumes** rel: TargetTerm TP R[\[LT\]<TRel>] TargetTerm TQ
- **Shows** (TP, TQ) ∈ TRel
  - **Using** rel
  - by (simp add: indRelLT.simps)

**Lemma (in encoding) indRelT-to-TRel:**

- **Fixes** TRel :: (procT × procT) set
- **And** TP TQ :: 'procT
- **Assumes** rel: TargetTerm TP R[\[T\]<TRel>] TargetTerm TQ
- **Shows** (TP, TQ) ∈ TRel
  - **Using** rel
  - by (simp add: indRelT.simps)

If the preorders indRelRTPO, indRelLTPO, or the equivalence indRelTEQ contain a pair of terms, then the pair of target terms that is related to these two terms is also related by the reflexive and transitive closure of the considered target term relation.

**Lemma (in encoding) indRelRTPO-to-TRel:**

- **Fixes** TRel :: (procT × procT) set
- **And** P Q :: (procS, 'procT) Proc
- **Assumes** rel: P ≤[\[RT\]<TRel>] Q
- **Shows** ∀SP SQ. SP ∈ S P ∧ SQ ∈ S Q → SP = SQ
  - **And** ∀SP TQ. SP ∈ S P ∧ TQ ∈ T Q → ([SP], TQ) ∈ (TRel ∪ {(T1, T2). ∃S. T1 = [S] ∧ T2 = [S]})^+
  - **And** ∀TP SQ. TP ∈ T P ∧ SQ ∈ S Q → False
  - **And** ∀TP TQ. TP ∈ T P ∧ TQ ∈ T Q → (TP, TQ) ∈ TRel^+
- **Proof –**
  - **Have** refTRel: ∀S. ([S], [S]) ∈ TRel ∪ {(T1, T2). ∃S. T1 = [S] ∧ T2 = [S]}
  - by auto
  - **From** rel show ∀SP SQ. SP ∈ S P ∧ SQ ∈ S Q → SP = SQ
    - **And** ∀SP TQ. SP ∈ S P ∧ TQ ∈ T Q → ([SP], TQ) ∈ (TRel ∪ {(T1, T2). ∃S. T1 = [S] ∧ T2 = [S]})^+
    - **And** ∀TP SQ. TP ∈ T P ∧ SQ ∈ S Q → False
    - **And** ∀TP TQ. TP ∈ T P ∧ TQ ∈ T Q → (TP, TQ) ∈ TRel^+
- **Proof induct**
  - **Case** (encR S)
    - **Show** ∀SP SQ. SP ∈ S SourceTerm S ∧ SQ ∈ S TargetTerm ([S]) → SP = SQ
    - **And** ∀TP SQ. TP ∈ T SourceTerm S ∧ SQ ∈ S TargetTerm ([S]) → False
    - **And** ∀TP TQ. TP ∈ T SourceTerm S ∧ TQ ∈ T TargetTerm ([S]) → (TP, TQ) ∈ TRel^+
    - by simp-all
  - **From** refTRel show ∀SP SQ. SP ∈ S SourceTerm S ∧ TQ ∈ T TargetTerm ([S]) → ([SP], TQ) ∈ (TRel ∪ {(T1, T2). ∃S. T1 = [S] ∧ T2 = [S]})^+
    - by blast
- **Next**
  - **Case** (source S)
    - **Show** ∀SP SQ. SP ∈ S SourceTerm S ∧ SQ ∈ S SourceTerm S → SP = SQ

87
by simp

show \( \forall SP \ TQ. \ SP \in S \ SourceTerm \ S \land \ TQ \in T \ SourceTerm \ S \)
\[ \rightarrow (\{SP\}, TQ) \in (TRel \cup \{(T1, T2)\}, \exists S. \ T1 = [S] \land T2 = [S])^+ \]

and \( \forall TP \ TQ. \ TP \in T \ SourceTerm \ S \land SQ \in S \ SourceTerm \ S \rightarrow False \)

and \( \forall TP TQ. \ TP \in T \ SourceTerm \ S \land TQ \in T \ SourceTerm \ S \rightarrow (TP, TQ) \in TRel^+ \)

by simp-all

next

case \((target \ T1 \ T2)\)

show \( \forall SP SQ. \ SP \in S \ TargetTerm \ T1 \land SQ \in S \ TargetTerm \ T2 \rightarrow SP = SQ \)

and \( \forall SP TQ. \ SP \in S \ TargetTerm \ T1 \land TQ \in T \ TargetTerm \ T2 \)
\[ \rightarrow (\{SP\}, TQ) \in (TRel \cup \{(T1, T2)\}, \exists S. \ T1 = [S] \land T2 = [S])^+ \]

and \( \forall TP SQ. \ TP \in T \ TargetTerm \ T1 \land SQ \in S \ TargetTerm \ T2 \rightarrow False \)

by simp-all

assume \((T1, T2) \in TRel\)

thus \( \forall TP TQ. \ TP \in T \ TargetTerm \ T1 \land TQ \in T \ TargetTerm \ T2 \rightarrow (TP, TQ) \in TRel^+ \)

by simp

next

case \((trans \ P \ Q \ R)\)

assume \(A1: \forall SP SQ. \ SP \in S \ P \land SQ \in S \ Q \rightarrow SP = SQ \)

and \(A2: \forall SP TQ. \ SP \in S \ P \land TQ \in T \ Q \)
\[ \rightarrow (\{SP\}, TQ) \in (TRel \cup \{(T1, T2)\}, \exists S. \ T1 = [S] \land T2 = [S])^+ \]

and \(A3: \forall TP SQ. \ TP \in T \ P \land SQ \in S \ Q \rightarrow False \)

and \(A4: \forall TP TQ. \ TP \in T \ P \land TQ \in T \ Q \rightarrow (TP, TQ) \in TRel^+ \)

and \(A5: \forall SQ SR. \ SQ \in S \ Q \land SR \in S \ R \rightarrow SQ = SR \)

and \(A6: \forall SQ TR. \ SQ \in S \ Q \land TR \in T \ R \)
\[ \rightarrow (\{SQ\}, TR) \in (TRel \cup \{(T1, T2)\}, \exists S. \ T1 = [S] \land T2 = [S])^+ \]

and \(A7: \forall TQ SQ. \ TQ \in T \ Q \land SQ \in S \ R \rightarrow False \)

and \(A8: \forall TQ TR. \ TQ \in T \ Q \land TR \in T \ R \rightarrow (TQ, TR) \in TRel^+ \)

show \( \forall SP SR. \ SP \in S \ P \land SR \in S \ R \rightarrow SP = SR \)

proof \((cases \ Q)\)

\(\text{case} \ (SourceTerm \ SQ)\)

assume \(SQ \in S \ Q \)

with \(A1 \ A5 \ \text{show} \ \forall SP SR. \ SP \in S \ P \land SR \in S \ R \rightarrow SP = SR \)

by blast

next

case \((TargetTerm \ TQ)\)

assume \(TQ \in T \ Q \)

with \(A7 \ \text{show} \ ?thesis \)

by blast

qed

show \( \forall SP TR. \ SP \in S \ P \land TR \in T \ R \)
\[ \rightarrow (\{SP\}, TR) \in (TRel \cup \{(T1, T2)\}, \exists S. \ T1 = [S] \land T2 = [S])^+ \]

proof \((cases \ Q)\)

\(\text{case} \ (SourceTerm \ SQ)\)

assume \(SQ \in S \ Q \)

with \(A1 \ A6 \ \text{show} \ ?thesis \)

by blast

next

case \((TargetTerm \ TQ)\)

assume \(A9: \ TQ \in T \ Q \)

show \( \forall SP TR. \ SP \in S \ P \land TR \in T \ R \)
\[ \rightarrow (\{SP\}, TR) \in (TRel \cup \{(T1, T2)\}, \exists S. \ T1 = [S] \land T2 = [S])^+ \]

proof \(\text{clarify}\)

fix \(SP \ TR\)

assume \(SP \in S \ P\)

with \(A2 \ A9 \ \text{have} \ (\{SP\}, TQ) \in (TRel \cup \{(T1, T2)\}, \exists S. \ T1 = [S] \land T2 = [S])^+ \)

by simp

moreover assume \(TR \in T \ R\)

with \(A8 \ A9 \ \text{have} \ (TQ, TR) \in TRel^+ \)

by simp

hence \((TQ, TR) \in (TRel \cup \{(T1, T2)\}, \exists S. \ T1 = [S] \land T2 = [S])^+ \)
\begin{proof}\textit{induct}

\begin{enumerate}
\item \texttt{fix T2}
\item \texttt{assume (TQ, T2) \in TRel}
\item \texttt{thus (TQ, T2) \in (TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\})^+}
\item \texttt{by blast}
\end{enumerate}
\end{proof}

\begin{proof}(\textit{cases Q})
\begin{enumerate}
\item \texttt{case (SourceTerm SQ)}
\item \texttt{assume SQ \in S Q}
\item \texttt{with A3 show \?thesis}
\item \texttt{by blast}
\end{enumerate}
\end{proof}

\begin{proof}(\textit{cases Q})
\begin{enumerate}
\item \texttt{case (TargetTerm TQ)}
\item \texttt{assume TQ \in T Q}
\item \texttt{with A7 show \?thesis}
\item \texttt{by blast}
\end{enumerate}
\end{proof}

\begin{proof}(\textit{cases Q})
\begin{enumerate}
\item \texttt{case (TargetTerm TQ)}
\item \texttt{assume TQ \in T Q}
\item \texttt{with A4 A8 show \forall TP TR. TP \in T P \land TR \in T R \rightarrow (TP, TR) \in TRel^+}
\item \texttt{by auto}
\end{enumerate}
\end{proof}

\begin{proof}(\textit{encoding}) \textit{indRelLTPO-to-TRel:}
\begin{enumerate}
\item \texttt{fixes TRel :: ('procT x 'procT) set}
\item \texttt{and P Q :: ('procS, 'procT) Proc}
\item \texttt{assumes rel: P \leq [1]LT< TRel> Q}
\item \texttt{shows \forall SP SQ. SP \in S P \land SQ \in S Q \rightarrow SP = SQ}
\item \texttt{and \forall SP TQ. SP \in S P \land TQ \in T Q \rightarrow False}
\item \texttt{and \forall TP SQ. TP \in T P \land SQ \in S Q}
\item \texttt{\rightarrow (TP, [SQ]) \in (TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\})^+}
\item \texttt{and \forall TP TQ. TP \in T P \land TQ \in T Q \rightarrow (TP, TQ) \in TRel^+}
\end{enumerate}
\end{proof}

\texttt{lemma (in encoding) \textit{indRelLTPO-to-TRel:}}

\begin{enumerate}
\item \texttt{fixes TRel :: ('procT x 'procT) set}
\item \texttt{and P Q :: ('procS, 'procT) Proc}
\item \texttt{assumes rel: P \leq [1]LT< TRel> Q}
\item \texttt{shows \forall SP SQ. SP \in S P \land SQ \in S Q \rightarrow SP = SQ}
\item \texttt{and \forall SP TQ. SP \in S P \land TQ \in T Q \rightarrow False}
\item \texttt{and \forall TP SQ. TP \in T P \land SQ \in S Q}
\item \texttt{\rightarrow (TP, [SQ]) \in (TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\})^+}
\end{enumerate}
and \(\forall TP \ TQ. TP \in T \land TQ \in Q \rightarrow (TP, TQ) \in TRel^+\)

**proof** \textit{induct}

**case** (encL \(S\))

\[\begin{align*}
\text{show} & \forall SP \ SQ. SP \in S \text{ TargetTerm } ([S]) \land SQ \in S \text{ SourceTerm } S \rightarrow SP = SQ \\
\text{and} & \forall SP \ TQ. SP \in S \text{ TargetTerm } ([S]) \land TQ \in T \text{ SourceTerm } S \rightarrow False \\
\text{and} & \forall TP \ TQ. TP \in T \text{ TargetTerm } ([S]) \land TQ \in T \text{ SourceTerm } S \rightarrow (TP, TQ) \in TRel^+
\end{align*}\]

by \textit{simp-all}

from \textit{reflTRel show} \(\forall TP \ SQ. TP \in T \text{ TargetTerm } ([S]) \land SQ \in S \text{ SourceTerm } S \rightarrow (TP, [SQ]) \in (TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\})^+\)

by \textit{blast}

**next**

**case** (source \(S\))

\[\begin{align*}
\text{show} & \forall SP \ SQ. SP \in S \text{ SourceTerm } S \land SQ \in S \text{ SourceTerm } S \rightarrow SP = SQ \\
\text{by} & \textit{simp}
\end{align*}\]

\[\begin{align*}
\text{show} & \forall SP \ TQ. SP \in S \text{ SourceTerm } S \land TQ \in T \text{ SourceTerm } S \rightarrow False \\
\text{and} & \forall TP \ SQ. TP \in T \text{ SourceTerm } S \land SQ \in S \text{ SourceTerm } S \\
\text{by} & \textit{simp-all}
\end{align*}\]

**next**

**case** (target \(T1 \ T2\))

\[\begin{align*}
\text{show} & \forall SP \ SQ. SP \in S \text{ TargetTerm } T1 \land SQ \in S \text{ TargetTerm } T2 \rightarrow SP = SQ \\
\text{and} & \forall SP \ TQ. SP \in S \text{ TargetTerm } T1 \land TQ \in T \text{ TargetTerm } T2 \rightarrow False \\
\text{and} & \forall TP \ SQ. TP \in T \text{ TargetTerm } T1 \land SQ \in S \text{ TargetTerm } T2 \\
\text{by} & \textit{simp-all}
\end{align*}\]

**assume** \((T1, T2) \in TRel\)

**thus** \(\forall TP \ TQ. TP \in T \text{ TargetTerm } T1 \land TQ \in T \text{ TargetTerm } T2 \rightarrow (TP, TQ) \in TRel^+\)

by \textit{simp}

**next**

**case** (trans \(P \ Q \ R\))

**assume** \((A1: \forall SP \ SQ. SP \in S \land SQ \in S \rightarrow SP = SQ)

\[\begin{align*}
\text{and} & A2: \forall SP \ TQ. SP \in S \land TQ \in T \rightarrow False \\
\text{and} & A3: \forall TP \ SQ. TP \in T \land SQ \in S \\
\text{by} & \textit{simp-all}
\end{align*}\]

**A4: \forall TP \ TQ. TP \in T \land TQ \in T \rightarrow (TP, TQ) \in TRel^+\)

**A5: \forall SQ \ SR. SQ \in S \land SR \in T \rightarrow SQ = SR\)

**A6: \forall SQ \ TR. SQ \in S \land TR \in T \rightarrow False\)

**A7: \forall QT \ SR. QT \in T \land SR \in S \rightarrow QT = SR\)

**A8: \forall QT \ TR. QT \in T \land TR \in T \rightarrow (QT, TR) \in TRel^+\)

**show** \(\forall SP \ SR. SP \in S \land SR \in S \rightarrow SP = SR\)

**proof** \textit{(cases \(Q\))}

**case** (SourceTerm \(SQ\))

**assume** \(SQ \in S \)

**with** \(A5 \) **show** \(\forall SP \ SR. SP \in S \land SR \in S \rightarrow SP = SR\)

by \textit{blast}

**next**

**case** (TargetTerm \(TQ\))

**assume** \(TQ \in T \)

**with** \(A2 \) **show** \(?thesis\)

by \textit{blast}

**qed**

**show** \(\forall SP \ TR. SP \in S \land TR \in T \rightarrow False\)

**proof** \textit{(cases \(Q\))}

**case** (SourceTerm \(SQ\))

**assume** \(SQ \in S \)

**with** \(A6 \) **show** \(?thesis\)

by \textit{blast}

**next**

**case** (TargetTerm \(TQ\))
assume $TQ \in T \ Q$
with $A2$ show $\%thesis$
  by blast
qed
show $\forall TP \ sr. TP \in T \ P \land SR \in S \ R$
  $\longrightarrow (TP, [SR]) \in (TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\}^+$
proof (cases $Q$)
  case (SourceTerm $SQ$)
  assume $SQ \in S \ Q$
  with $A3$ $A5$ show $\forall TP \ sr. TP \in T \ P \land SR \in S \ R$
    $\longrightarrow (TP, [SR]) \in (TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\}^+$
  by blast
next
case (TargetTerm $TQ$
  assume $A9: TQ \in T \ Q$
  show $\forall TP \ sr. TP \in T \ P \land SR \in S \ R$
    $\longrightarrow (TP, [SR]) \in (TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\}^+$
proof clarify
  fix $TP \ sr$
  assume $TP \in T \ P$
  with $A4$ $A9$ have $(TP, TQ) \in TRel^+$
    by simp
  hence $(TP, TQ) \in (TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\}^+$
proof induct
  fix $T2$
  assume $(TP, T2) \in TRel$
  thus $(TP, T2) \in (TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\}^+$
    by blast
next
case (step $T2$ $T3$)
  assume $(TP, T2) \in (TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\}^+$
  moreover assume $(T2, T3) \in TRel$
  hence $(T2, T3) \in (TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\}^+$
    by blast
  ultimately show $(TP, T3) \in (TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\}^+$
    by simp
qed
moreover assume $SR \in S \ R$
with $A7$ $A9$ have $(TQ, [SR]) \in (TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\}^+$
  by simp
ultimately show $(TP, [SR]) \in (TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\}^+$
  by simp
qed
qed
show $\forall TP \ TR. TP \in T \ P \land TR \in T \ R$ $\longrightarrow (TP, TR) \in TRel^+$
proof (cases $Q$)
  case (SourceTerm $SQ$
  assume $SQ \in S \ Q$
  with $A6$ show $\%thesis$
    by blast
next
case (TargetTerm $TQ$
  assume $TQ \in T \ Q$
  with $A4$ $A8$ show $\forall TP \ TR. TP \in T \ P \land TR \in T \ R$ $\longrightarrow (TP, TR) \in TRel^+$
    by auto
qed
qed

lemma (in encoding) $\%indRel\%\%TEQ\%to\%\%TRel$
  fixes $TRel :: (\%procT \times \%procT\%)$ set
and \( P \) \( Q \) :: (\('\text{proc}S', \text{'}\text{proc}T\)) \( \text{Proc} \)

assumes \( \text{rel} : P \sim T \rightarrow T < T\text{Rel} > Q \)

shows \( \forall SP SQ. \; SP \in S \land SQ \in S \; Q \)

\[ \rightarrow ([SP], [SQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

and \( \forall SP TQ. \; SP \in S \land TQ \in T \; Q \)

\[ \rightarrow ([SP], [TQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

and \( \forall TP SQ. \; TP \in T \land SP \in S \; Q \)

\[ \rightarrow (TP, [SQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

and \( \forall TP TQ. \; TP \in T \land TQ \in T \; Q \)

\[ \rightarrow (TP, [TQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

proof

have \( \text{reflTRel} \; : \forall S. \; ([S], [S]) \in T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\} \)

by auto

from \( \text{rel} \) show \( \forall SP SQ. \; SP \in S \land SQ \in S \; Q \)

\[ \rightarrow ([SP], [SQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

and \( \forall SP TQ. \; SP \in S \land TQ \in T \; Q \)

\[ \rightarrow ([SP], [TQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

and \( \forall TP SQ. \; TP \in T \land SP \in S \; Q \)

\[ \rightarrow (TP, [SQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

and \( \forall TP TQ. \; TP \in T \land TQ \in T \; Q \)

\[ \rightarrow (TP, [TQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

proof

induct

case \( \text{encR} \; S \)

show \( \forall SP SQ. \; SP \in S \land SQ \in S \; S \; \text{SourceTerm} \)

\[ \rightarrow ([SP], [SQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

and \( \forall TP SQ. \; TP \in T \land SP \in S \; S \; \text{SourceTerm} \)

\[ \rightarrow (TP, [SQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

and \( \forall TP TQ. \; TP \in T \land TP \in T \; \text{SourceTerm} \)

\[ \rightarrow (TP, [TQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

by simp+

from \( \text{reflTRel} \) show \( \forall SP TQ. \; SP \in S \land TQ \in T \; \text{TargetTerm} \)

\[ \rightarrow ([SP], [TQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

by blast

next

case \( \text{encL} \; S \)

show \( \forall SP SQ. \; SP \in S \land SQ \in S \; \text{SourceTerm} \)

\[ \rightarrow ([SP], [SQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

and \( \forall TP SQ. \; TP \in T \land SP \in S \; S \; \text{SourceTerm} \)

\[ \rightarrow (TP, [SQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

and \( \forall TP TQ. \; TP \in T \land TP \in T \; \text{SourceTerm} \)

\[ \rightarrow (TP, [TQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

by simp+

from \( \text{reflTRel} \) show \( \forall TP SQ. \; TP \in T \land SQ \in S \; \text{SourceTerm} \)

\[ \rightarrow (TP, [SQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

by blast

next

case \( \text{target} \; T1 \land T2 \)

show \( \forall SP SQ. \; SP \in S \land \text{TargetTerm} \; T1 \land SQ \in S \; \text{TargetTerm} \; T2 \)

\[ \rightarrow ([SP], [SQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

and \( \forall TP SQ. \; SP \in S \land \text{TargetTerm} \; T1 \land TQ \in T \; \text{TargetTerm} \; T2 \)

\[ \rightarrow ([SP], [TQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

and \( \forall TP TQ. \; TP \in T \land \text{TargetTerm} \; T1 \land SQ \in S \; \text{TargetTerm} \; T2 \)

\[ \rightarrow (TP, [SQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

by simp+

assume \( (T1, T2) \in \text{TRel} \)

thus \( \forall TP TQ. \; TP \in T \land \text{TargetTerm} \; T1 \land TQ \in T \; \text{TargetTerm} \; T2 \)

\[ \rightarrow (TP, [TQ]) \in (T\text{Rel} \cup \{(T1, T2). \exists S. \; T1 = [S] \land T2 = [S]\})^+ \]

by blast

next

case \( \text{trans} \; P \land Q \land R \)

assume \( A1 : \forall SP SQ. \; SP \in S \land SQ \in S \; Q \)
∀ \show{TQ} \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+

\and A2: \forall SP TQ. SP \in S P \land TQ \in T Q
\rightarrow (\[SP\], TQ) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+

\and A3: \forall TP SQ. TP \in T P \land SQ \in S Q
\rightarrow (TP, [SQ]) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+

\and A4: \forall TP TQ. TP \in T P \land TQ \in T Q
\rightarrow (TP, TQ) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+

\and A5: \forall SQ SR. SQ \in S Q \land SR \in S R
\rightarrow ([SQ], [SR]) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+

\and A6: \forall SQ TR. SQ \in S Q \land TR \in T R
\rightarrow ([SQ], TR) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+

\and A7: \forall TQ SR. TQ \in T Q \land SR \in S R
\rightarrow (TQ, [SR]) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+

\and A8: \forall TQ TR. TQ \in T Q \land TR \in T R
\rightarrow (TQ, TR) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+

\show \forall SP SR. SP \in S P \land SR \in S R
\rightarrow ([SP], [SR]) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+$

\proof (cases Q)
\begin{itemize}
\item case (SourceTerm SQ)
  \assum SQ \in S Q
  \with A1 A5 show \thesis
    \by auto
\end{itemize}

\qed

\show \forall SP TR. SP \in S P \land TR \in T R \rightarrow ([SP], TR) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+

\proof (cases Q)
\begin{itemize}
\item case (SourceTerm SQ)
  \assum SQ \in S Q
  \with A1 A6 show \thesis
    \by auto
\end{itemize}

\qed

\show \forall TP SR. TP \in T P \land SR \in S R \rightarrow (TP, [SR]) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+

\proof (cases Q)
\begin{itemize}
\item case (SourceTerm SQ)
  \assum SQ \in S Q
  \with A3 A5 show \thesis
    \by auto
\end{itemize}

\qed

\show \forall TP TR. TP \in T P \land TR \in T R \rightarrow (TP, TR) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+

\proof (cases Q)
\begin{itemize}
\item case (SourceTerm SQ)
  \assum SQ \in S Q
  \with A3 A6 show \thesis
    \by auto
\end{itemize}

\qed

\show \forall SP TR. SP \in S P \land TR \in T Q
\rightarrow ([SP], TR) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+$

\proof (cases Q)
\begin{itemize}
\item case (SourceTerm SQ)
  \assum SQ \in S Q
  \with A4 A7 show \thesis
    \by auto
\end{itemize}

\qed

\show \forall TP TR. TP \in T P \land TR \in T Q
\rightarrow (TP, TR) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+

\proof (cases Q)
\begin{itemize}
\item case (SourceTerm SQ)
  \assum SQ \in S Q
  \with A4 A8 show \thesis
    \by auto
\end{itemize}

\qed

\show \forall TP TR. TP \in T P \land TR \in T \rightarrow (TP, TR) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+$

\proof (cases Q)
\begin{itemize}
\item case (SourceTerm SQ)
  \assum SQ \in S Q
  \with A6 show \thesis
    \by auto
\end{itemize}

\qed

\show \forall TP TR. TP \in T P \land TR \in T \rightarrow (TP, TR) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+$

\proof (cases Q)
\begin{itemize}
\item case (SourceTerm SQ)
  \assum SQ \in S Q
  \with A5 show \thesis
    \by auto
\end{itemize}

\qed

\show \forall SP TR. SP \in S P \land TR \in T \rightarrow ([SP], TR) \in (TRel \cup \{(T1, T2), \exists S. T1 = [S] \land T2 = [S])^+

\proof (cases Q)
\begin{itemize}
\item case (SourceTerm SQ)
  \assum SQ \in S Q
  \with A2 show \thesis
    \by auto
\end{itemize}

\qed
with \( A_4, A_8 \) show \(?thesis\)
by auto
qed
qed
qed

lemma (in encoding) trans-closure-of-TRel-refl-cond:
  fixes \( TRel :: ('procT \times 'procT) \text{ set} \)
  and \( TP \ TQ :: 'procT \)
  assumes \( (TP, TQ) \in (TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\})^+ \)
  shows \( (TP, TQ) \in TRel^* \)
  using \text{assms}
proof induct
fix \( TQ \)
assume \( (TP, TQ) \in TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\} \)
thus \( (TP, TQ) \in TRel^* \)
  by auto
next
  case (step \( TQ \ TR \))
  assume \( (TP, TQ) \in TRel \cup \{(T1, T2). \exists S. T1 = [S] \land T2 = [S]\} \)
  hence \( (TQ, TR) \in TRel^* \)
  by blast
  ultimately show \( (TP, TR) \in TRel^* \)
  by simp
qed

Note that if \( \text{indRelRTPO} \) relates a source term \( S \) to a target term \( T \), then the translation of \( S \) is equal to \( T \) or \( \text{indRelRTPO} \) also relates the translation of \( S \) to \( T \).

lemma (in encoding) \( \text{indRelRTPO-relates-source-target} \):
  fixes \( TRel :: ('procT \times 'procT) \text{ set} \)
  and \( S :: 'procS \)
  and \( T :: 'procT \)
  assumes \( \text{pair}: \text{SourceTerm} \ S \subseteq \RT<T\text{R}\text{rel} \text{TargetTerm} \ T \)
  shows \( (\text{TargetTerm} ([S]), \text{TargetTerm} T) \in (\text{indRelRTPO} \ TRel)^= \)
proof
  from \text{pair} have \( ([S], T) \in TRel^* \)
  using \text{indRelRTPO-to-TRel2} \text{where} TRel=\text{TRel} \text{trans-closure-of-TRel-refl-cond}
  by simp
  hence \( [S] = T \lor ([S], T) \in TRel^+ \)
  using rtrancl-eq-or-trancl[of \([S]\ \ T \ TRel\]
  by blast
  moreover have \( [S] = T \implies (\text{TargetTerm} ([S]), \text{TargetTerm} T) \in (\text{indRelRTPO} \ TRel)^= \)
  by simp
  moreover
  have \( ([S], T) \in TRel^+ \implies (\text{TargetTerm} ([S]), \text{TargetTerm} T) \in (\text{indRelRTPO} \ TRel)^= \)
  using transitive-closure-of-TRel-to-\text{indRelRTPO} \text{where} TRel=\text{TRel}
  by simp
  ultimately show \( (\text{TargetTerm} ([S]), \text{TargetTerm} T) \in (\text{indRelRTPO} \ TRel)^= \)
  by blast
qed

If \( \text{indRelRTPO} \), \( \text{indRelLTPO} \), or \( \text{indRelTPO} \) preserves barbs then so does the corresponding target term relation.

lemma (in encoding-wrt-barbs) \( \text{rel-with-target-impl-TRel-preserves-barbs} \):
  fixes \( TRel :: ('procT \times 'procT) \text{ set} \)
  and \( \text{Rel} :: ('procS, 'procT) \text{Proc} \times ('procS, 'procT) \text{Proc} \text{ set} \)
  assumes \( \text{preservation}: \text{rel-preserves-barbs Rel} \ (\text{STCalWB SWB TWB}) \)
  and \( \text{targetInRel}: \forall T1 \ T2. \ (T1, T2) \in TRel \implies (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel} \)
  shows \( \text{rel-preserves-barbs TRel TWB} \)
proof clarify
fix TP TQ a
assume (TP, TQ) ∈ TRel
with targetInRel have (TargetTerm TP, TargetTerm TQ) ∈ Rel
by blast
moreover assume TP ↓<TWB>a
hence TargetTerm TP ↓a
by simp
ultimately have TargetTerm TQ ↓a
using preservation preservation-of-barbs-in-barbed-encoding[where Rel=Rel]
by blast
thus TQ ↓<TWB>a
by simp
qed

lemma (in encoding-wrt-barbs) indRelRTPO-impl-TRel-preserves-barbs:
fixes TRel :: (′procT × ′procT) set
assumes preservation: rel-preserves-barbs (indRelRTPO TRel) (STCalWB SWB TWB)
shows rel-preserves-barbs TRel TWB
using preservation
rel-with-target-impl-TRel-preserves-barbs[where Rel=indRelRTPO TRel and TRel=TRel]
by ( simp add: indRelRTPO.target )

lemma (in encoding-wrt-barbs) indRelLTPO-impl-TRel-preserves-barbs:
fixes TRel :: (′procT × ′procT) set
assumes preservation: rel-preserves-barbs (indRelLTPO TRel) (STCalWB SWB TWB)
shows rel-preserves-barbs TRel TWB
using preservation
rel-with-target-impl-TRel-preserves-barbs[where Rel=indRelLTPO TRel and TRel=TRel]
by ( simp add: indRelLTPO.target )

lemma (in encoding-wrt-barbs) indRelTEQ-impl-TRel-preserves-barbs:
fixes TRel :: (′procT × ′procT) set
assumes preservation: rel-preserves-barbs (indRelTEQ TRel) (STCalWB SWB TWB)
shows rel-preserves-barbs TRel TWB
using preservation
rel-with-target-impl-TRel-preserves-barbs[where Rel=indRelTEQ TRel and TRel=TRel]
by ( simp add: indRelTEQ.target )

lemma (in encoding-wrt-barbs) rel-with-target-impl-TRel-weakly-preserves-barbs:
fixes TRel :: (′procT × ′procT) set
and Rel :: (′procS, ′procT) Proc × (′procS, ′procT) Proc set
assumes preservation: rel-weakly-preserves-barbs Rel (STCalWB SWB TWB)
and targetInRel: ∀ T1 T2. (T1, T2) ∈ TRel → (TargetTerm T1, TargetTerm T2) ∈ Rel
shows rel-weakly-preserves-barbs TRel TWB
proof clarify
fix TP TQ a TP'
assume (TP, TQ) ∈ TRel
with targetInRel have (TargetTerm TP, TargetTerm TQ) ∈ Rel
by blast
moreover assume TP —>(Calculus TWB)* TP' and TP' ↓<TWB>a
hence TargetTerm TP' ↓a
by blast
ultimately have TargetTerm TQ ↓a
using preservation weak-preservation-of-barbs-in-barbed-encoding[where Rel=Rel]
by blast
thus TQ ↓<TWB>a
by simp
qed

lemma (in encoding-wrt-barbs) indRelRTPO-impl-TRel-weakly-preserves-barbs:
fixes \( TRel :: (\texttt{procT} \times \texttt{procT}) \) set
assumes preservation: rel-weakly-preserves-barbs (indRelRTPO \( TRel \) ) (STCalWB SWB TWB)
shows rel-weakly-preserves-barbs \( TRel \) TWB
  using preservation rel-with-target-impl-TRel-weakly-preserves-barbs
  \( \text{where} \quad \text{Rel}=\text{indRelRTPO } TRel \text{ and } TRel=\text{TRel} \)
by (simp add: indRelRTPO.target)

lemma (in encoding-wrt-barbs) indRelLTPO-impl-TRel-weakly-preserves-barbs:
fixes \( TRel :: (\texttt{procT} \times \texttt{procT}) \) set
assumes preservation: rel-weakly-preserves-barbs (indRelLTPO \( TRel \) ) (STCalWB SWB TWB)
shows rel-weakly-preserves-barbs \( TRel \) TWB
  using preservation rel-with-target-impl-TRel-weakly-preserves-barbs
  \( \text{where} \quad \text{Rel}=\text{indRelLTPO } TRel \text{ and } TRel=\text{TRel} \)
by (simp add: indRelLTPO.target)

lemma (in encoding-wrt-barbs) indRelTEQ-impl-TRel-weakly-preserves-barbs:
fixes \( TRel :: (\texttt{procT} \times \texttt{procT}) \) set
assumes preservation: rel-weakly-preserves-barbs (indRelTEQ \( TRel \) ) (STCalWB SWB TWB)
shows rel-weakly-preserves-barbs \( TRel \) TWB
  using preservation rel-with-target-impl-TRel-weakly-preserves-barbs
  \( \text{where} \quad \text{Rel}=\text{indRelTEQ } TRel \text{ and } TRel=\text{TRel} \)
by (simp add: indRelTEQ.target)

If \( \text{indRelRTPO} \), \( \text{indRelLTPO} \), or \( \text{indRelTPO} \) reflects barbs then so does the corresponding target term relation.

lemma (in encoding-wrt-barbs) rel-with-target-impl-TRel-reflects-barbs:
fixes \( TRel :: (\texttt{procT} \times \texttt{procT}) \) set
and \( \text{Rel} :: (\texttt{procS}, \texttt{procT}) \) \( \text{Proc} \times (\texttt{procS}, \texttt{procT}) \) \( \text{Proc} \) set
assumes reflection: rel-reflects-barbs \( \text{Rel} \) (STCalWB SWB TWB)
and targetInRel: \( \forall \ T1 \ T2. \ (T1, T2) \in TRel \rightarrow (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel} \)
shows rel-reflects-barbs \( TRel \) TWB
proof clarify
fix \( TP \ TQ \ a \)
assume \( (TP, TQ) \in TRel \)
with targetInRel have \( (\text{TargetTerm } TP, \text{TargetTerm } TQ) \in \text{Rel} \)
  by blast
moreover assume \( TQ \downarrow \langle \text{TWB} > a \)
hence \( \text{TargetTerm } TQ \downarrow . a \)
  by simp
ultimately have \( \text{TargetTerm } TP \downarrow . a \)
  using reflection reflection-of-barbs-in-barbed-encoding
  \( \text{where} \quad \text{Rel}=\text{Rel} \)
  by blast
thus \( TP \downarrow \langle \text{TWB} > a \)
  by simp
qed

lemma (in encoding-wrt-barbs) indRelRTPO-impl-TRel-reflects-barbs:
fixes \( TRel :: (\texttt{procT} \times \texttt{procT}) \) set
assumes reflection: rel-reflects-barbs \( \text{indRelRTPO } TRel \) (STCalWB SWB TWB)
shows rel-reflects-barbs \( TRel \) TWB
  using reflection
  rel-with-target-impl-TRel-reflects-barbs
  \( \text{where} \quad \text{Rel}=\text{indRelRTPO } TRel \text{ and } TRel=\text{TRel} \)
by (simp add: indRelRTPO.target)

lemma (in encoding-wrt-barbs) indRelLTPO-impl-TRel-reflects-barbs:
fixes \( TRel :: (\texttt{procT} \times \texttt{procT}) \) set
assumes reflection: rel-reflects-barbs \( \text{indRelLTPO } TRel \) (STCalWB SWB TWB)
shows rel-reflects-barbs \( TRel \) TWB
  using reflection
  rel-with-target-impl-TRel-reflects-barbs
  \( \text{where} \quad \text{Rel}=\text{indRelLTPO } TRel \text{ and } TRel=\text{TRel} \)
by \((\text{simp add: indRelLTPO.target})\)

**Lemma** (in encoding-wrt-barbs) \(\text{indRelTEQ-impl-TRel-reflects-barbs}\):

- \(\text{fixes } TRel :: (\text{'proc} T \times \text{'proc} T) \set\)
- \(\text{assumes reflection: rel-reflects-barbs (indRelTEQ TRel) (STCalWB SWB TWB)}\)
- \(\text{shows rel-reflects-barbs TRel TWB using reflection}\)
- \(\text{rel-with-target-impl-TRel-reflects-barbs[where } Rel = \text{indRelTEQ TRel and TRel = TRel]}\)
- \(\text{by (simp add: indRelTEQ.target)}\)

**Lemma** (in encoding-wrt-barbs) \(\text{rel-with-target-impl-TRel-weakly-reflects-barbs}\):

- \(\text{fixes } TRel :: (\text{'proc} T \times \text{'proc} T) \set\)
- \(\text{and Rel :: (('procS, 'procT) Proc \times ('procS, 'procT) Proc) set}\)
- \(\text{assumes reflection: rel-weakly-reflects-barbs Rel (STCalWB SWB TWB)}\)
- \(\text{and targetInRel: } \forall T1 T2. (T1, T2) \in TRel \rightarrow (\text{TargetTerm } T1, \text{TargetTerm } T2) \in Rel\)
- \(\text{shows rel-weakly-reflects-barbs TRel TWB}\)

**Proof** clarify

- fix \(TP, TQ, a, TQ'\)
- assume \((TP, TQ) \in TRel\)
- with targetInRel have \((\text{TargetTerm } TP, \text{TargetTerm } TQ) \in Rel\)
- by blast
- moreover assume \(TQ \rightarrow (\text{Calculus TWB})^* TQ'\) and \(TQ' \downarrow <TWB>a\)
- hence \(\text{TargetTerm } TQ\downarrow .a\)
- by blast
- ultimately have \(\text{TargetTerm } TP\downarrow .a\)
- using reflection weak-reflection-of-barbs-in-barbed-encoding[where \(Rel = Rel\)]
- by blast
- thus \(TP\downarrow <TWB>a\)
- by simp

**Qed**

**Lemma** (in encoding-wrt-barbs) \(\text{indRelRTPO-impl-TRel-weakly-reflects-barbs}\):

- \(\text{fixes } TRel :: (\text{'proc} T \times \text{'proc} T) \set\)
- \(\text{assumes reflection: rel-weakly-reflects-barbs (indRelRTPO TRel) (STCalWB SWB TWB)}\)
- \(\text{shows rel-weakly-reflects-barbs TRel TWB using reflection rel-with-target-impl-TRel-weakly-reflects-barbs[where } Rel = \text{indRelRTPO TRel and TRel = TRel]}\)
- \(\text{by (simp add: indRelRTPO.target)}\)

**Lemma** (in encoding-wrt-barbs) \(\text{indRelLTPO-impl-TRel-weakly-reflects-barbs}\):

- \(\text{fixes } TRel :: (\text{'proc} T \times \text{'proc} T) \set\)
- \(\text{assumes reflection: rel-weakly-reflects-barbs (indRelLTPO TRel) (STCalWB SWB TWB)}\)
- \(\text{shows rel-weakly-reflects-barbs TRel TWB using reflection rel-with-target-impl-TRel-weakly-reflects-barbs[where } Rel = \text{indRelLTPO TRel and TRel = TRel]}\)
- \(\text{by (simp add: indRelLTPO.target)}\)

**Lemma** (in encoding-wrt-barbs) \(\text{indRelTEQ-impl-TRel-weakly-reflects-barbs}\):

- \(\text{fixes } TRel :: (\text{'proc} T \times \text{'proc} T) \set\)
- \(\text{assumes reflection: rel-weakly-reflects-barbs (indRelTEQ TRel) (STCalWB SWB TWB)}\)
- \(\text{shows rel-weakly-reflects-barbs TRel TWB using reflection rel-with-target-impl-TRel-weakly-reflects-barbs[where } Rel = \text{indRelTEQ TRel and TRel = TRel]}\)
- \(\text{by (simp add: indRelTEQ.target)}\)

If \(\text{indRelRTPO}, \text{indRelLTPO}, \text{or indRelTEQ respects barbs} \) then so does the corresponding target term relation.

**Lemma** (in encoding-wrt-barbs) \(\text{indRelRTPO-impl-TRel-respects-barbs}\):

- \(\text{fixes } TRel :: (\text{'proc} T \times \text{'proc} T) \set\)
- \(\text{assumes respection: rel-respects-barbs (indRelRTPO TRel) (STCalWB SWB TWB)}\)
- \(\text{shows rel-respects-barbs TRel TWB using reflection rel-with-target-impl-TRel-respects-barbs[where } Rel = \text{indRelRTPO TRel and TRel = TRel]}\)
- \(\text{by (simp add: indRelRTPO.target)}\)
clarify proof lemma

If \text{indRelRTPO}, \text{indRelLTPO}, or \text{indRelTEQ} is a simulation then so is the corresponding target term relation.

\text{lemma (in encoding-wrt-barbs) indRelLTPO-impl-TRel-respects-barbs:}
\begin{itemize}
  \item fixes \text{TRel} :: (‘procT × ‘procT) set
  \item assumes \text{respection: rel-respects-barbs \( \text{indRelLTPO TRel} \) \( \text{STCalWB SWB TWB} \)}
  \item shows \text{rel-respects-barbs \TRel \ TWB}
    \begin{itemize}
      \item using \text{respection indRelLTPO-impl-TRel-preserves-barbs[where \ TRel=TRel]}
      \begin{itemize}
        \item \text{indRelLTPO-impl-TRel-reflection-barbs[where \ TRel=TRel]}
      \end{itemize}
    \end{itemize}
\end{itemize}
by blast

\text{lemma (in encoding-wrt-barbs) indRelTEQ-impl-TRel-respects-barbs:}
\begin{itemize}
  \item fixes \text{TRel} :: (‘procT × ‘procT) set
  \item assumes \text{respection: rel-respects-barbs \( \text{indRelTEQ TRel} \) \( \text{STCalWB SWB TWB} \)}
  \item shows \text{rel-respects-barbs \TRel \ TWB}
    \begin{itemize}
      \item using \text{respection indRelTEQ-impl-TRel-preserves-barbs[where \ TRel=TRel]}
      \begin{itemize}
        \item \text{indRelTEQ-impl-TRel-reflection-barbs[where \ TRel=TRel]}
      \end{itemize}
    \end{itemize}
\end{itemize}
by blast

\text{lemma (in encoding-wrt-barbs) indRelRTPO-impl-TRel-weakly-respects-barbs:}
\begin{itemize}
  \item fixes \text{TRel} :: (‘procT × ‘procT) set
  \item assumes \text{respection: rel-weakly-respects-barbs \( \text{indRelRTPO TRel} \) \( \text{STCalWB SWB TWB} \)}
  \item shows \text{rel-weakly-respects-barbs \TRel \ TWB}
    \begin{itemize}
      \item using \text{respection indRelRTPO-impl-TRel-weakly-preserves-barbs[where \ TRel=TRel]}
      \begin{itemize}
        \item \text{indRelRTPO-impl-TRel-weakly-reflection-barbs[where \ TRel=TRel]}
      \end{itemize}
    \end{itemize}
\end{itemize}
by blast

\text{lemma (in encoding-wrt-barbs) indRelLTPO-impl-TRel-weakly-respects-barbs:}
\begin{itemize}
  \item fixes \text{TRel} :: (‘procT × ‘procT) set
  \item assumes \text{respection: rel-weakly-respects-barbs \( \text{indRelLTPO TRel} \) \( \text{STCalWB SWB TWB} \)}
  \item shows \text{rel-weakly-respects-barbs \TRel \ TWB}
    \begin{itemize}
      \item using \text{respection indRelLTPO-impl-TRel-weakly-preserves-barbs[where \ TRel=TRel]}
      \begin{itemize}
        \item \text{indRelLTPO-impl-TRel-weakly-reflection-barbs[where \ TRel=TRel]}
      \end{itemize}
    \end{itemize}
\end{itemize}
by blast

\text{lemma (in encoding-wrt-barbs) indRelTEQ-impl-TRel-weakly-respects-barbs:}
\begin{itemize}
  \item fixes \text{TRel} :: (‘procT × ‘procT) set
  \item assumes \text{respection: rel-weakly-respects-barbs \( \text{indRelTEQ TRel} \) \( \text{STCalWB SWB TWB} \)}
  \item shows \text{rel-weakly-respects-barbs \TRel \ TWB}
    \begin{itemize}
      \item using \text{respection indRelTEQ-impl-TRel-weakly-preserves-barbs[where \ TRel=TRel]}
      \begin{itemize}
        \item \text{indRelTEQ-impl-TRel-weakly-reflection-barbs[where \ TRel=TRel]}
      \end{itemize}
    \end{itemize}
\end{itemize}
by blast

If \text{indRelRTPO}, \text{indRelLTPO}, or \text{indRelTEQ} is a simulation then so is the corresponding target term relation.

\text{lemma (in encoding) rel-with-target-impl-transC-TRel-is-weak-reduction-simulation:}
\begin{itemize}
  \item fixes \text{TRel} :: (‘procT × ‘procT) set
  \item assumes \text{respection: rel-with-target-impl-transC-TRel-is-weak-reduction-simulation \( \text{STCal Source Target} \)}
  \item and \text{target: \forall \ T1 \ T2. (T1, T2) ∈ \TRel \rightarrow \text{(TargetTerm T1, TargetTerm T2) ∈ Rel}}
  \item and \text{map: \forall \ T1 \ T2. (TargetTerm T1, TargetTerm T2) ∈ \Rel \rightarrow \text{(T1, T2) ∈ \TRel}^+}
  \item shows \text{weak-reduction-simulation \( \TRel^+ \) Target}
\end{itemize}
\text{proof clarify}
\begin{itemize}
  \item fix \text{TP \ TQ \ TP'}
  \item assume \(\text{(TP, TQ)} \in \TRel^+ \text{ and } \text{TP} \rightarrow\text{Target* TP'}\)
  \item thus \(\exists \text{TQ'}. \text{TQ} \rightarrow\text{Target* TQ'} \land \text{(TP', TQ') ∈ \TRel}^+\)
  \item proof (induct arbitrary: TP')
  \begin{itemize}
    \item fix \text{TQ \ TP'}
  \end{itemize}
\end{itemize}
assume \((TP, TQ) \in TRel\)

with target have \((\text{TargetTerm TP}, \text{TargetTerm TQ}) \in \text{Rel}\)

by simp

moreover assume \(TP \rightarrow \text{Target}* TP'\)

hence \(\text{TargetTerm TP} \rightarrow (\text{STCal Source Target})* (\text{TargetTerm TP}')\)

by \((\text{simp add: STCal-steps})\)

ultimately obtain \(Q'\) where \(A2: \text{TargetTerm TQ} \rightarrow (\text{STCal Source Target})* Q'\)

and \(A3: (\text{TargetTerm TP}', Q') \in \text{Rel}\)

using \(\text{sim}\)

by blast

from \(A2\) obtain \(TQ'\) where \(A4: TQ \rightarrow \text{Target}* TQ'\) and \(A5: TQ' \in T Q'\)

by \((\text{auto simp add: STCal-steps})\)

from \(A3\ A5\ trel\) have \((TP', TQ') \in TRel^+\)

by simp

with \(A4\) show \(\exists TQ', TQ \rightarrow \text{Target}* TQ' \land (TP', TQ') \in TRel^+\)

by blast

next

case \((\text{step TQ TR})\)

assume \(TP \rightarrow \text{Target}* TP'\)

and \(\land TP', TP \rightarrow \text{Target}* TP' \Rightarrow \exists TQ'. TQ \rightarrow \text{Target}* TQ' \land (TP', TQ') \in TRel^+\)

from this obtain \(TQ'\) where \(B1: TQ \rightarrow \text{Target}* TQ'\) and \(B2: (TP', TQ') \in TRel^+\)

by blast

assume \((TQ, TR) \in TRel\)

with target have \((\text{TargetTerm TQ}, \text{TargetTerm TR}) \in \text{Rel}\)

by simp

moreover from \(B1\) have \(\text{TargetTerm TQ} \rightarrow (\text{STCal Source Target})* (\text{TargetTerm TQ}')\)

by \((\text{simp add: STCal-steps})\)

ultimately obtain \(R'\) where \(B3: \text{TargetTerm TR} \rightarrow (\text{STCal Source Target})* R'\)

and \(B4: (\text{TargetTerm TQ}', R') \in \text{Rel}\)

using \(\text{sim}\)

by blast

from \(B3\) obtain \(TR'\) where \(B5: TR' \in T R'\) and \(B6: TR \rightarrow \text{Target}* TR'\)

by \((\text{auto simp add: STCal-steps})\)

from \(B4\ B5\ trel\) have \((TQ', TR') \in TRel^+\)

by simp

with \(B2\) have \((TP', TR') \in TRel^+\)

by simp

with \(B6\) show \(\exists TR', TR \rightarrow \text{Target}* TR' \land (TP', TR') \in TRel^+\)

by blast

qed

qed

lemma \((\text{in encoding})\) \text{indRelRTPO-impl-TRel-is-weak-reduction-simulation}:

fixes \(TRel :: (\text{procT} \times \text{procT})\) set

assumes \(\text{sim}: \text{weak-reduction-simulation} (\text{indRelRTPO TRel}) (\text{STCal Source Target})\)

shows \(\text{weak-reduction-simulation} (\text{TRel}^+)\) Target

using \(\text{sim}\) \text{indRelRTPO.target}[where \(TRel=TRel\)] \text{indRelRTPO-to-TRel}(4)[where \(TRel=TRel\)]

\text{rel-with-target-impl-transC-TRel-is-weak-reduction-simulation}[where \(Rel=\text{indRelRTPO TRel}\) and \(TRel=TRel\)]

by blast

lemma \((\text{in encoding})\) \text{indRelLTPO-impl-TRel-is-weak-reduction-simulation}:

fixes \(TRel :: (\text{procT} \times \text{procT})\) set

assumes \(\text{sim}: \text{weak-reduction-simulation} (\text{indRelLTPO TRel}) (\text{STCal Source Target})\)

shows \(\text{weak-reduction-simulation} (\text{TRel}^+)\) Target

using \(\text{sim}\) \text{indRelLTPO.target}[where \(TRel=TRel\)] \text{indRelLTPO-to-TRel}(4)[where \(TRel=TRel\)]

\text{rel-with-target-impl-transC-TRel-is-weak-reduction-simulation}[where \(Rel=\text{indRelLTPO TRel}\) and \(TRel=TRel\)]

by blast

lemma \((\text{in encoding})\) \text{rel-with-target-impl-transC-TRel-is-weak-reduction-simulation-rev}:
fixes \( TRel \) :: \( \langle \text{proc}T \times \text{proc}T \rangle \) set 
and \( \text{Rel} \) :: \( \langle \langle \text{proc}S, \text{proc}T \rangle \text{ Proc} \times \langle \text{proc}S, \text{proc}T \rangle \text{ Proc} \rangle \) set 

assumes sim: weak-reduction-simulation \((\text{Rel}^{-1}) \) \((\text{STCal Source Target}) \)

and target: \( \forall T_1 T_2. \ (T_1, T_2) \in TRel \rightarrow (\text{TargetTerm} T_1, \text{TargetTerm} T_2) \in \text{Rel} \)

and trel: \( \forall T_1 T_2. \ (\text{TargetTerm} T_1, \text{TargetTerm} T_2) \in \text{Rel} \rightarrow (T_1, T_2) \in TRel^+ \)

shows weak-reduction-simulation \((TRel^+)^{-1} \) \((\text{Target}) \)

proof clarify

fix \( TP \ TQ \ TP' \)

assume \( (TQ, TP) \in TRel^+ \)

moreover assume \( TP \rightarrow\text{Target}* TP' \)

ultimately show \( \exists TQ'. \ TQ \rightarrow\text{Target}* TQ' \land (TP', TQ') \in (TRel^+)^{-1} \)

proof (induct arbitrary: \( TP' \))

fix \( TP \ TP' \)

assume \( (TQ, TP) \in TRel \)

with target have \( (\text{TargetTerm} TP, \text{TargetTerm} TQ) \in \text{Rel}^{-1} \)

by simp

moreover assume \( TP \rightarrow\text{Target}* TP' \)

hence \( \text{TargetTerm} TP \rightarrow (\text{STCal Source Target})* (\text{TargetTerm} TP') \)

by \( \text{simp add: STCal-steps} \)

ultimately obtain \( Q' \) where \( A2: \text{TargetTerm} TQ \rightarrow (\text{STCal Source Target})* Q' \)

and \( A3: (\text{TargetTerm} TP', Q') \in \text{Rel}^{-1} \)

using sim

by blast

from \( A2 \) obtain \( TQ' \) where \( A4: TQ \rightarrow\text{Target}* TQ' \) and \( A5: TQ' \in T Q' \)

by \( \text{auto simp add: STCal-steps(2)} \)

from \( A3 A5 \) trel have \( (TP', TQ') \in (TRel^+)^{-1} \)

by simp

with \( A4 \) show \( \exists TQ'. \ TQ \rightarrow\text{Target}* TQ' \land (TP', TQ') \in (TRel^+)^{-1} \)

by blast

next

case \( \text{step TR TP TP'} \)

assume \( TP \rightarrow\text{Target}* TP' \)

hence \( \text{TargetTerm} TP \rightarrow (\text{STCal Source Target})* (\text{TargetTerm} TP') \)

by \( \text{simp add: STCal-steps} \)

moreover assume \( (TR, TP) \in TRel \)

with target have \( (\text{TargetTerm} TP, \text{TargetTerm} TR) \in \text{Rel}^{-1} \)

by simp

ultimately obtain \( R' \) where \( B1: \text{TargetTerm} TR \rightarrow (\text{STCal Source Target})* R' \)

and \( B2: (\text{TargetTerm} TP', R') \in \text{Rel}^{-1} \)

using sim

by blast

from \( B1 \) obtain \( TR' \) where \( B3: TR' \in T R' \) and \( B4: TR \rightarrow\text{Target}* TR' \)

by \( \text{auto simp add: STCal-steps} \)

assume \( \forall TR'. \ TR \rightarrow\text{Target}* TR' \rightarrow \exists TQ'. \ TQ \rightarrow\text{Target}* TQ' \land (TR', TQ') \in (TRel^+)^{-1} \)

with \( B4 \) obtain \( TQ' \) where \( B5: TQ \rightarrow\text{Target}* TQ' \) and \( B6: (TR', TQ') \in (TRel^+)^{-1} \)

by blast

from \( B6 \) have \( (TQ', TR') \in TRel^+ \)

by simp

moreover from \( B2 B3 \) trel have \( (TR', TP') \in TRel^+ \)

by simp

ultimately have \( (TP', TQ') \in (TRel^+)^{-1} \)

by simp

with \( B5 \) show \( \exists TQ'. \ TQ \rightarrow\text{Target}* TQ' \land (TP', TQ') \in (TRel^+)^{-1} \)

by blast

qed

qed

lemma (in encoding) indRelRTPO-impl-TRel-is-weak-reduction-simulation-rev:

fixes \( TRel \) :: \( \langle \text{proc}T \times \text{proc}T \rangle \) set 

assumes sim: weak-reduction-simulation \( ((\text{indRelRTPO} \ TRel)^{-1}) \) \((\text{STCal Source Target}) \)

shows weak-reduction-simulation \( ((TRel^+)^{-1} \) \((\text{Target}) \)

100
using sim indRelRTPO.target[where TRel=TRel] indRelRTPO-to-TRel[4][where TRel=TRel]
rel-with-target-impl-transC-TRel-is-weak-reduction-simulation-rev[where
Rel=indRelRTPO TRel and TRel=TRel]
by blast

lemma (in encoding) indRelLTPO-impl-TRel-is-weak-reduction-simulation-rev:
fixes TRel :: ('procT × 'procT) set
assumes sim: weak-reduction-simulation ((indRelLTPO TRel)⁻¹) (STCal Source Target)
shows weak-reduction-simulation ((TRel⁺)⁻¹) Target
using sim indRelLTPO.target[where TRel=TRel] indRelLTPO-to-TRel[4][where TRel=TRel]
rel-with-target-impl-transC-TRel-is-weak-reduction-simulation-rev[where
Rel=indRelLTPO TRel and TRel=TRel]
by blast

lemma (in encoding) rel-with-target-impl-refC-transC-TRel-is-weak-reduction-simulation:
fixes TRel :: ('procT × 'procT) set
and Rel :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) set
assumes sim: weak-reduction-simulation Rel (STCal Source Target)
and target: ∀ T1 T2. (T1, T2) ∈ TRel → (TargetTerm T1, TargetTerm T2) ∈ Rel
and trel: ∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel → (T1, T2) ∈ TRel⁺
shows weak-reduction-simulation (TRel⁺) Target
proof clarify
fix TP TQ TP'
assume (TP, TQ) ∈ TRel⁺ and TP → Target TP'
thus ∃ TQ'. TQ → Target TQ' ∧ (TP', TQ') ∈ TRel⁺
proof (induct arbitrary: TP')
fix TP'
assume TP → Target TP'
moreover have (TP', TP') ∈ TRel⁺
by simp
ultimately show ∃ TQ'. TP → Target TQ' ∧ (TP', TQ') ∈ TRel⁺
by blast
next
case (step TQ TR)
assume TP → Target TR
and ∃ TP'. TP → Target TP' → ∃ TQ'. TQ → Target TQ' ∧ (TP', TQ') ∈ TRel⁺
from this obtain TQ' where B1: TQ → Target TQ' and B2: (TP', TQ') ∈ TRel⁺
by blast
assume (TQ, TR) ∈ TRel
with target have (TargetTerm TQ, TargetTerm TR) ∈ Rel
by simp
moreover from B1 have TargetTerm TQ → STCal Source Target)* (TargetTerm TQ')
by (simp add: STCal-steps)
ultimately obtain R' where B3: TargetTerm TR → STCal Source Target)* R'
and B4: (TargetTerm TQ', R') ∈ Rel
using sim
by blast
from B3 obtain TR' where B5: TR' ∈ T R' and B6: TR → Target TR'
by (auto simp add: STCal-steps)
from B4 B5 trrel have (TQ', TR') ∈ TRel⁺
by simp
with B6 show ∃ TR'. TR → Target TR' ∧ (TP', TR') ∈ TRel⁺
by blast
qed
qed

lemma (in encoding) indRelTEQ-impl-TRel-is-weak-reduction-simulation:
fixes TRel :: ('procT × 'procT) set
assumes sim: weak-reduction-simulation (indRelTEQ TRel) (STCal Source Target)
shows weak-reduction-simulation (TRel\^+) Target
using sim indRelTEQ.target[\textbf{where} TRel=TRel] indRelTEQ-to-TRel(4)[\textbf{where} TRel=TRel]
trans-closure-of-TRel-refl-cond
rel-with-target-impl-reflC-transC-TRel-is-weak-reduction-simulation[\textbf{where}]
  Rel=\textit{indRelTEQ TRel and TRel=TRel}]
by blast

lemma (in encoding) rel-with-target-impl-transC-TRel-is-strong-reduction-simulation:
fixes TRel :: ('procT × 'procT) set
  \textit{and} Rel :: ('procS, 'procT) Proc × ('procS, 'procT) Proc) set
assumes sim: \textit{strong-reduction-simulation} Rel (STCal Source Target)
  \textit{and target:} \(\forall T1 T2. (T1, T2) \in \text{TRel} \rightarrow (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel}\)
  \textit{and trel:} \(\forall T1 T2. (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel} \rightarrow (T1, T2) \in \text{TRel}^+\)
shows \textit{strong-reduction-simulation} (TRel\^+) Target
proof clarify
fix TP TQ TP'
assume (TP, TQ) ∈ TRel\^+ and TP ⧦ Target TP'
thus ∃ TQ'. TQ ⧦ Target TQ' ∧ (TP', TQ') ∈ TRel\^+
proof (induct arbitrary: TP')
fix TP TP'
assume (TP, TQ) ∈ TRel
with target have (TargetTerm TP, TargetTerm TQ) ∈ Rel
  by simp
moreover assume TP ⧦ Target TP'
hence TargetTerm TP ⧦ \textit{(STCal Source Target)} (TargetTerm TP')
  by (simp add: STCal-step)
ultimately obtain Q' where A2: TargetTerm TQ ⧦ \textit{(STCal Source Target)} Q'
  \textit{and} A3: (TargetTerm TP', Q') ∈ Rel
    using sim
    by blast
from A2 obtain TQ' where A4: TQ ⧦ Target TQ' and A5: TQ' ∈ T Q'
    by (auto simp add: STCal-step)
from A3 A5 trel have (TP', TQ') ∈ TRel\^+
    by simp
with A4 show ∃ TQ'. TQ ⧦ Target TQ' ∧ (TP', TQ') ∈ TRel\^+
    by blast

next
case (step TQ TR)
assume TP ⧦ Target TP'
  \textit{and} \(\land TP', TP ⧦ Target TP' \implies \exists TQ'. TQ ⧦ Target TQ' ∧ (TP', TQ') \in TRel^+\)
from this obtain TQ' where B1: TQ ⧦ Target TQ' \textit{and} B2: (TP', TQ') ∈ TRel\^+
  by blast
assume (TQ, TR) ∈ TRel
with target have (TargetTerm TQ, TargetTerm TR) ∈ Rel
  by simp
moreover from B1 have TargetTerm TQ ⧦ \textit{(STCal Source Target)} (TargetTerm TQ')
    by (simp add: STCal-step)
ultimately obtain R' where B3: TargetTerm TR ⧦ \textit{(STCal Source Target)} R'
  \textit{and} B4: (TargetTerm TQ', R') ∈ Rel
    using sim
    by blast
from B3 obtain TR' where B5: TR' ∈ T R' \textit{and} B6: TR ⧦ Target TR'
    by (auto simp add: STCal-step)
from B4 B5 trel have (TQ', TR') ∈ TRel\^+
    by simp
with B2 have (TP', TR') ∈ TRel\^+
    by simp
with B6 show ∃ TR'. TR ⧦ Target TR' ∧ (TP', TR') ∈ TRel\^+
    by blast
qed
qed
lemma (in encoding) \( \text{indRelRTPO-impl-TRel-is-strong-reduction-simulation} \):

\[
\begin{align*}
\text{fixes} & \quad \text{TRel} :: (\text{'procT} \times \text{'procT}) \text{ set} \\
\text{assumes} & \quad \text{sim} : \text{strong-reduction-simulation} (\text{indRelRTPO TRel}) (\text{STCal Source Target}) \\
\text{shows} & \quad \text{strong-reduction-simulation} (\text{TRel}^+) \text{ Target} \\
\text{using} & \quad \text{sim indRelRTPO.target[where TRel=TRel]} \quad \text{indRelRTPO-to-TRel(4)[where TRel=TRel]} \\
& \quad \text{rel-with-target-impl-transC-TRel-is-strong-reduction-simulation[where Rel=indRelRTPO TRel and TRel=TRel]} \\
\text{by} & \quad \text{blast}
\end{align*}
\]

lemma (in encoding) \( \text{indRelLTPO-impl-TRel-is-strong-reduction-simulation} \):

\[
\begin{align*}
\text{fixes} & \quad \text{TRel} :: (\text{'procT} \times \text{'procT}) \text{ set} \\
\text{assumes} & \quad \text{sim} : \text{strong-reduction-simulation} (\text{indRelLTPO TRel}) (\text{STCal Source Target}) \\
\text{shows} & \quad \text{strong-reduction-simulation} (\text{TRel}^+) \text{ Target} \\
\text{using} & \quad \text{sim indRelLTPO.target[where TRel=TRel]} \quad \text{indRelLTPO-to-TRel(4)[where TRel=TRel]} \\
& \quad \text{rel-with-target-impl-transC-TRel-is-strong-reduction-simulation[where Rel=indRelLTPO TRel and TRel=TRel]} \\
\text{by} & \quad \text{blast}
\end{align*}
\]

lemma (in encoding) \( \text{rel-with-target-impl-transC-TRel-is-strong-reduction-simulation-rev} \):

\[
\begin{align*}
\text{fixes} & \quad \text{TP} \quad \text{TQ} \quad \text{TP}' \\
& \quad \text{Rel} :: ((\text{'procS}, \text{'procT}) \text{ Proc} \times (\text{'procS}, \text{'procT}) \text{ Proc}) \text{ set} \\
\text{assumes} & \quad \text{sim} : \text{strong-reduction-simulation} (\text{Rel}^{-1}) (\text{STCal Source Target}) \\
& \quad \text{and target: } \forall \ T1 \ T2. (\text{T1}, \text{T2}) \in \text{Rel} \rightarrow (\text{TargetTerm T1}, \text{TargetTerm T2}) \in \text{Rel} \\
& \quad \text{and trel: } \forall \ T1 \ T2. (\text{TargetTerm T1}, \text{TargetTerm T2}) \in \text{Rel} \rightarrow (\text{T1}, \text{T2}) \in \text{TRel}^+ \\
\text{shows} & \quad \text{strong-reduction-simulation} ((\text{TRel}^+)^{-1}) \text{ Target} \\
\text{proof} & \quad \text{clarify} \\
& \quad \text{fix TP TP' TQ} \\
& \quad \text{assume (TQ, TP) } \in \text{TRel}^+ \\
& \quad \text{moreover assume TP } \mapsto \text{Target TP'} \\
& \quad \text{ultimately show } \exists \ TQ'. \ TQ \mapsto \text{Target} \quad (\text{TP'}, \text{TQ'}) \in (\text{TRel}^+)^{-1} \\
\text{proof (induct arbitrary: TP')} \\
& \quad \text{fix TP TP'} \\
& \quad \text{assume (TQ, TP) } \in \text{TRel} \\
& \quad \text{with target have (TargetTerm TP, TargetTerm TQ) } \in \text{Rel}^{-1} \\
& \quad \text{by simp} \\
& \quad \text{moreover assume TP } \mapsto \text{Target TP'} \\
& \quad \text{hence TargetTerm TP } \mapsto (\text{STCal Source Target}) (\text{TargetTerm TP'}) \\
& \quad \text{by (simp add: STCal-step)} \\
& \quad \text{ultimately obtain Q' where A2: TargetTerm TP } \mapsto (\text{STCal Source Target}) \quad Q' \\
& \quad \text{and A3: (TargetTerm TP', Q') } \in \text{Rel}^{-1} \\
& \quad \text{using sim} \\
& \quad \text{by blast} \\
& \quad \text{from A2 obtain TQ' where A4: TQ } \mapsto \text{Target} \quad \text{TQ'} \quad \text{and A5: TQ'} \in \text{T Q'} \\
& \quad \text{by (auto simp add: STCal-step(2))} \\
& \quad \text{from A3 A5 trel have (TP', TQ') } \in (\text{TRel}^+)^{-1} \\
& \quad \text{by simp} \\
& \quad \text{with A4 show } \exists \ TQ'. \ TQ \mapsto \text{Target} \quad (\text{TP'}, \text{TQ'}) \in (\text{TRel}^+)^{-1} \\
& \quad \text{by blast} \\
\text{next} \\
& \quad \text{case (step TP TR TR')} \\
& \quad \text{assume (TP, TR) } \in \text{TRel} \\
& \quad \text{with target have (TargetTerm TP, TargetTerm TR) } \in \text{Rel} \\
& \quad \text{by simp} \\
& \quad \text{moreover assume TR } \mapsto \text{Target TR'} \\
& \quad \text{hence TargetTerm TR } \mapsto (\text{STCal Source Target}) (\text{TargetTerm TR'}) \\
& \quad \text{by (simp add: STCal-step)} \\
& \quad \text{ultimately obtain P' where B1: TargetTerm TP } \mapsto (\text{STCal Source Target}) \quad P' \\
& \quad \text{and B2: (P', TargetTerm TR') } \in \text{Rel} \\
& \quad \text{using sim} \\
& \quad \text{by blast} 
\end{align*}
\]
from $B_1$ obtain $TP'$ where $B_3$: $TP' \in T P'$ and $B_4$: $TP \longrightarrow Target \ TP'$
 by (auto simp add: STCal-step)
assume $\bigwedge TP', TP \longrightarrow Target \ TP' \implies \exists TQ', TQ \longrightarrow Target \ TQ' \land (TP', TQ') \in (TRel^+)^{-1}$
with $B_4$ obtain $TQ'$ where $B_5$: $TQ \longrightarrow Target \ TQ'$ and $B_6$: $(TP', TQ') \in (TRel^+)^{-1}$
 by blast
from $B_2$ $B_3$ trel have $(TP', TR') \in TRel^+$
 by simp
with $B_6$ have $(TR', TQ') \in (TRel^+)^{-1}$
 by simp
with $B_5$ show $\exists TQ', TQ \longrightarrow Target \ TQ' \land (TR', TQ') \in (TRel^+)^{-1}$
 by blast
qed
qed

lemma (in encoding) $\text{indRelRTPO-impl-TRel-is-strong-reduction-simulation-rev}$:
 fixes $TRel :: (\text{proc}T \times \text{proc}T)$ set
 assumes sim: $\text{strong-reduction-simulation} \ ((\text{indRelRTPO} \ TRel)^{-1}) \ (\text{STCal Source Target})$
 shows $\text{strong-reduction-simulation} \ ((TRel^+)^{-1}) \ Target$
 using sim indRelRTPO.target[where $TRel=TRel$] indRelRTPO-to-TRel(4)[where $TRel=TRel$]
 rel-with-target-impl-transC-TRel-is-strong-reduction-simulation-rev[where $TRel=\text{indRelRTPO} \ TRel$ and $TRel=TRel$]
 by blast
lemma (in encoding) $\text{indRelLTOPO-impl-TRel-is-strong-reduction-simulation-rev}$:
 fixes $TRel :: (\text{proc}T \times \text{proc}T)$ set
 assumes sim: $\text{strong-reduction-simulation} \ ((\text{indRelLTOPO} \ TRel)^{-1}) \ (\text{STCal Source Target})$
 shows $\text{strong-reduction-simulation} \ ((TRel^+)^{-1}) \ Target$
 using sim indRelLTOPO.target[where $TRel=TRel$] indRelLTOPO-to-TRel(4)[where $TRel=TRel$]
 rel-with-target-impl-transC-TRel-is-strong-reduction-simulation-rev[where $TRel=\text{indRelLTOPO} \ TRel$ and $TRel=TRel$]
 by blast
lemma (in encoding) $\text{rel-with-target-impl-reflC-transC-TRel-is-strong-reduction-simulation}$:
 fixes $TRel :: (\text{proc}T \times \text{proc}T)$ set
 and $Rel :: ((\text{proc}S, \text{proc}T) \text{Proc} \times (\text{proc}S, \text{proc}T) \text{Proc})$ set
 assumes sim: $\text{strong-reduction-simulation} \ Rel \ (\text{STCal Source Target})$
 and target: $\forall T1. T2. (T1, T2) \in TRel \longrightarrow (\text{TargetTerm} \ T1, \text{TargetTerm} \ T2) \in Rel$
 and trel: $\forall T1. T2. (\text{TargetTerm} \ T1, \text{TargetTerm} \ T2) \in Rel$
 
$\longrightarrow (T1, T2) \in TRel^*$
 shows $\text{strong-reduction-simulation} \ (TRel^*) \ Target$
proof clarify
fix $TP \ TQ \ TP'$
assume $(TP, TQ) \in TRel^*$ and $TP \longrightarrow Target \ TP'$
thus $\exists TQ', TQ \longrightarrow Target \ TQ' \land (TP', TQ') \in TRel^*$

proof ($\text{induct arbitrary:} \ TP'$)
fix $TP'$
assume $TP \longrightarrow Target \ TP'$
moreover have $(TP', TP') \in TRel^*$
 by simp
ultimately show $\exists TQ', TP \longrightarrow Target \ TQ' \land (TP', TQ') \in TRel^*$
 by blast
next
case (step $TQ \ TR \ TP'$)
assume $TP \longrightarrow Target \ TP'$
 and $\bigwedge TP', TP \longrightarrow Target \ TP' \implies \exists TQ', TQ \longrightarrow Target \ TQ' \land (TP', TQ') \in TRel^*$
from this obtain $TQ'$ where $B_1$: $TQ \longrightarrow Target \ TQ'$ and $B_2$: $(TP', TQ') \in TRel^*$
 by blast
assume $(TQ, TR) \in TRel$
with target have $(\text{TargetTerm} \ TQ, \text{TargetTerm} \ TR) \in Rel$
 by simp
moreover from $B_1$ have $\text{TargetTerm} \ TQ \longrightarrow (\text{STCal Source Target}) \ (\text{TargetTerm} \ TQ')$
by (simp add: STCal-step)
ultimately obtain $R'$ where $B3$: TargetTerm $TR \mapsto (STCal Source Target) R'$
and $B4$: (TargetTerm $TQ', R') \in Rel$

using sim
by blast
from $B3$ obtain $TR'$ where $B5$: $TR' \in T R'$ and $B6$: $TR \mapsto \text{Target TR'}$
by (auto simp add: STCal-step)
from $B4 \ B5 \ \text{trrel have} \ (TQ', TR') \in TRel$
by simp
with $B2$ have $(TP', TR') \in TRel$
by simp
with $B6$ show $\exists \ TR'. \ TR \mapsto \text{Target TR'} \land (TP', TR') \in TRel$
by blast
qed

lemma (in encoding) indRelTEQ-impl-TRel-is-strong-reduction-simulation:
fixes $TRel :: ('procT \times 'procT) \text{ set}$
assumes sim: strong-reduction-simulation ($indRelTEQ TRel \ (STCal Source Target)$
shows strong-reduction-simulation ($TRel^+$) Target
using sim indRelTEQ-target[where $TRel=TRel$]
indRelTEQ-to-TRel(4)[where $TRel=TRel$
trans-closure-of-TRel-refl-cond
rel-with-target-impl-refC-transC-TRel-is-strong-reduction-simulation[where
Rel=indRelTEQ TRel and $TRel=TRel$]
by blast

lemma (in encoding-wrt-barbs) indRelRTPO-impl-TRel-is-weak-barbed-simulation:
fixes $TRel :: ('procT \times 'procT) \text{ set}$
assumes sim: weak-barbed-simulation ($indRelRTPO TRel \ (STCalWB SWB TWB)$
shows weak-barbed-simulation ($TRel^+$) TWB
proof
from sim show weak-reduction-simulation ($TRel^+$) (Calculus TWB)
using indRelRTPO-impl-TRel-is-weak-reduction-simulation[where $TRel=TRel$
weak-preservation-of-barbs-and-closures(2)[where Rel=TRel and CWB=TWB]
by blast
next
from sim show rel-weakly-preserve-barbs ($TRel^+$) TWB
using indRelRTPO-impl-TRel-weakly-preserve-barbs[where $TRel=TRel$
weak-preservation-of-barbs-and-closures(2)[where Rel=TRel and CWB=TWB]
by blast
qed

lemma (in encoding-wrt-barbs) indRelLTPO-impl-TRel-is-weak-barbed-simulation:
fixes $TRel :: ('procT \times 'procT) \text{ set}$
assumes sim: weak-barbed-simulation ($indRelLTPO TRel \ (STCalWB SWB TWB)$
shows weak-barbed-simulation ($TRel^+$) TWB
proof
from sim show weak-reduction-simulation ($TRel^+$) (Calculus TWB)
using indRelLTPO-impl-TRel-is-weak-reduction-simulation[where $TRel=TRel$
by (simp add: STCalWB-def calS calT)
next
from sim show rel-weakly-preserve-barbs ($TRel^+$) TWB
using indRelLTPO-impl-TRel-weakly-preserve-barbs[where $TRel=TRel$
weak-preservation-of-barbs-and-closures(2)[where Rel=TRel and CWB=TWB]
by blast
qed

lemma (in encoding-wrt-barbs) indRelTEQ-impl-TRel-is-weak-barbed-simulation:
fixes $TRel :: ('procT \times 'procT) \text{ set}$
assumes sim: weak-barbed-simulation ($indRelTEQ TRel \ (STCalWB SWB TWB)$
shows weak-barbed-simulation ($TRel^+$) TWB
proof
from sim show weak-reduction-simulation $(TRel^*)$ (Calculus TWB) 
using indRelTEQ-impl-TRel-is-weak-reduction-simulation[where $TRel=TRel$]
by (simp add: STCalWB-def calS calT)

next
from sim show rel-weakly-preserves-barbs $(TRel^*)$ TWB 
using indRelTEQ-impl-TRel-weakly-preserves-barbs[where $TRel=TRel$] 
weak-preservation-of-barbs-and-closures(3)[where $Rel=TRel$ and $CWB=TWB$]
by blast

qed

lemma (in encoding-wrt-barbs) indRelRTPO-impl-TRel-is-strong-barbed-simulation:
fixes $TRel :: ('procT \times 'procT)$ set 
assumes sim: strong-barbed-simulation (indRelRTPO TRel) (STCalWB SWB TWB)
shows strong-barbed-simulation $(TRel^*)$ TWB
proof
from sim refl show strong-reduction-simulation $(TRel^*)$ (Calculus TWB) 
using indRelRTPO-impl-TRel-is-strong-reduction-simulation[where $TRel=TRel$]
by (simp add: STCalWB-def calS calT)
next
from sim show rel-preserves-barbs $(TRel^*)$ TWB 
using indRelRTPO-impl-TRel-preserves-barbs[where $TRel=TRel$] 
preservation-of-barbs-and-closures(2)[where $Rel=TRel$ and $CWB=TWB$]
by blast

qed

lemma (in encoding-wrt-barbs) indRelLTPO-impl-TRel-is-strong-barbed-simulation:
fixes $TRel :: ('procT \times 'procT)$ set 
assumes sim: strong-barbed-simulation (indRelLTPO TRel) (STCalWB SWB TWB)
shows strong-barbed-simulation $(TRel^*)$ TWB
proof
from sim refl show strong-reduction-simulation $(TRel^*)$ (Calculus TWB) 
using indRelLTPO-impl-TRel-is-strong-reduction-simulation[where $TRel=TRel$]
by (simp add: STCalWB-def calS calT)
next
from sim show rel-preserves-barbs $(TRel^*)$ TWB 
using indRelLTPO-impl-TRel-preserves-barbs[where $TRel=TRel$] 
preservation-of-barbs-and-closures(2)[where $Rel=TRel$ and $CWB=TWB$]
by blast

qed

lemma (in encoding-wrt-barbs) indRelTEQ-impl-TRel-is-strong-barbed-simulation:
fixes $TRel :: ('procT \times 'procT)$ set 
assumes sim: strong-barbed-simulation (indRelTEQ TRel) (STCalWB SWB TWB)
shows strong-barbed-simulation $(TRel^*)$ TWB
proof
from sim refl show strong-reduction-simulation $(TRel^*)$ (Calculus TWB) 
using indRelTEQ-impl-TRel-is-strong-reduction-simulation[where $TRel=TRel$]
by (simp add: STCalWB-def calS calT)
next
from sim show rel-preserves-barbs $(TRel^*)$ TWB 
using indRelTEQ-impl-TRel-preserves-barbs[where $TRel=TRel$] 
preservation-of-barbs-and-closures(3)[where $Rel=TRel$ and $CWB=TWB$]
by blast

qed

If indRelRTPO, indRelLTPO, or indRelTEQ is a contrasimulation then so is the corresponding target term relation.

lemma (in encoding) rel-with-target-impl-transC-TRel-is-weak-reduction-contrasimulation:
fixes $TRel :: ('procT \times 'procT)$ set 
and $Rel :: (('procS, 'procT) Proc \times ('procS, 'procT) Proc) set$
assumes conSim: weak-reduction-contrasimulation Rel (STCal Source Target)
and target: ∀ T1 T2. (T1, T2) ∈ TRel → (TargetTerm T1, TargetTerm T2) ∈ Rel
and trel: ∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel → (T1, T2) ∈ TRel+
shows weak-reduction-contrasimulation (TRel+) Target

proof clarify

fix TP TQ TP'
assume (TP, TQ) ∈ TRel+ and TP → Target* TP'
thus ∃ TQ'. TQ → Target* TQ' ∧ (TQ', TP') ∈ TRel+

proof (induct arbitrary: TP')

fix TQ TP'
assume (TP, TQ) ∈ TRel

with target have (TargetTerm TP, TargetTerm TQ) ∈ Rel
by simp

moreover assume TP → Target* TP'
hence TargetTerm TP ⟷ (STCal Source Target)* (TargetTerm TP')
by (simp add: STCal-steps)

ultimately obtain Q' where A2: TargetTerm TQ ⟷ (STCal Source Target)* Q'
and A3: (Q', TargetTerm TP') ∈ Rel

using conSim
by blast

from A2 obtain TQ' where A4: TQ ⟷ Target* TQ' and A5: TQ' ∈ T Q'
by (auto simp add: STCal-steps)

from A3 A5 trel have (TQ', TP') ∈ TRel+
by simp

with A4 show ∃ TQ'. TQ → Target* TQ' ∧ (TQ', TP') ∈ TRel+
by blast

next

case (step TQ TR)

assume TP → Target* TP'
and T Q'. TP → Target* TP' ⟷ ∃ TQ'. TQ → Target* TQ' ∧ (TQ', TP') ∈ TRel+

from this obtain TQ' where B1: TQ → Target* TQ' and B2: (TQ', TP') ∈ TRel+
by blast

assume (TQ, TR) ∈ TRel

with target have (TargetTerm TQ, TargetTerm TR) ∈ Rel
by simp

moreover from B1 have TargetTerm TQ → (STCal Source Target)* (TargetTerm TQ')
by (simp add: STCal-steps)

ultimately obtain R' where B3: TargetTerm TR → (STCal Source Target)* R'
and B4: (R', TargetTerm TQ') ∈ Rel

using conSim
by blast

from B3 obtain TR' where B5: TR' ∈ T R' and B6: TR → Target* TR'
by (auto simp add: STCal-steps)

from B4 B5 trel have (TR', TQ') ∈ TRel+
by simp

from this B2 have (TR', TP') ∈ TRel+
by simp

with B6 show ∃ TR'. TR → Target* TR' ∧ (TR', TP') ∈ TRel+
by blast

qed

qed

lemma (in encoding) indRelRTPO-impl-TRel-is-weak-reduction-contrasimulation:

fixes TRel :: ('procT × 'procT) set

assumes conSim: weak-reduction-contrasimulation (indRelRTPO TRel) (STCal Source Target)

shows weak-reduction-contrasimulation (TRel+) Target

using conSim indRelRTPO.target[where TRel=TRel] indRelRTPO-to-TRel(4)[where TRel=TRel]
rel-with-target-impl-transC-TRel-is-weak-reduction-contrasimulation[where Rel=indRelRTPO TRel and TRel=TRel]
by blast
lemma (in encoding) \textit{indRelLTPO-impl-TRel-is-weak-reduction-contrasimulation}:

\begin{itemize}
\item \textbf{fixes} \textit{TRel} :: \textit{\'(procT \times \'procT\') set}
\item \textbf{assumes} \textit{conSim}:: weak-reduction-contrasimulation (\textit{indRelLTPO TRel}) (\textit{STCal Source Target})
\item \textbf{shows} weak-reduction-contrasimulation (\textit{TRel\^\!*}) \textit{Target}
\item using \textit{conSim} \textit{indRelLTPO-target[\textbf{where} TRel=\textit{TRel}] indRelLTPO-to-\textit{TRel}(.)[\textbf{where} TRel=\textit{TRel}] rel-with-target-impl-transC-\textit{TRel-is-weak-reduction-contrasimulation}[\textbf{where} Rel=\textit{indRelLTPO TRel and TRel=\textit{TRel}]}
\item by \textit{blast}
\end{itemize}

lemma (in encoding) \textit{rel-with-target-impl-reflC-transC-\textit{TRel-is-weak-reduction-contrasimulation}}:

\begin{itemize}
\item \textbf{fixes} \textit{TRel} :: \textit{\'(procS \times \'procT\') Proc \times \textit{\'(procS, 'procT) Proc\') set}
\item \textbf{assumes} \textit{conSim}:: weak-reduction-contrasimulation \textit{Rel} (\textit{STCal Source Target})
\item and \textit{target}:: \forall \textit{T1 T2}. \textit{(T1, T2) \in TRel \rightarrow (TargetTerm T1, TargetTerm T2) \in Rel}
\item and \textit{trel}:: \forall \textit{T1 T2}. (\textit{TargetTerm T1, TargetTerm T2) \in Rel \rightarrow (T1, T2) \in TRel\^\!*}
\item \textbf{shows} weak-reduction-contrasimulation (\textit{TRel\^\!*}) \textit{Target}
\item \textbf{proof} \textit{clarify}
\item \textbf{fix} \textit{TP TQ TP\^\!*}
\item \textbf{assume} \textit{(TP, TQ) \in TRel\^\!* and TP \rightarrow Target\^\!* TP\^\!*}
\item \textbf{thus} \exists \textit{TQ\^\!*}. \textit{TQ \rightarrow Target\^\!* TQ\^\!* \land (TQ\^\!, TP\^\!) \in TRel\^\!*}
\item \textbf{proof} (\textit{induct arbitrary: \textit{TP\^\!*}})
\item \textbf{fix} \textit{TP\^\!*}
\item \textbf{assume} \textit{TP \rightarrow Target\^\!* TP\^\!*}
\item \textbf{moreover have} \textit{(TP\^\!, TP\^\!) \in TRel\^\!*}
\item \textbf{by} \textit{simp}
\item \textbf{ultimately show} \exists \textit{TQ\^\!*}. \textit{TP \rightarrow Target\^\!* TQ\^\!* \land (TQ\^\!, TP\^\!) \in TRel\^\!*}
\item \textbf{by} \textit{blast}
\item \textbf{next}
\item \textbf{case} (\textit{step TQ TR})
\item \textbf{assume} \textit{TP \rightarrow Target\^\!* TP\^\!*}
\item \textbf{and} \(\land TP\^\!.\) \textit{TP \rightarrow Target\^\!* TP\^\!* \Rightarrow \exists TQ\^\!.\ TQ \rightarrow Target\^\!* TQ\^\!* \land (TQ\^\!, TP\^\!) \in TRel\^\!*}
\item from \textit{this obtain} \textit{TQ\^\!* where} \textit{B1: TQ \rightarrow Target\^\!* TQ\^\!* and B2: (TQ\^\!, TP\^\!) \in TRel\^\!*}
\item \textbf{by} \textit{blast}
\item \textbf{assume} \textit{(TQ, TR) \in TRel}
\item with \textit{target have} \textit{(TargetTerm TQ, TargetTerm TR) \in Rel}
\item \textbf{by} \textit{simp}
\item \textbf{moreover from} \textit{B1 have} \textit{TargetTerm TQ \rightarrow (STCal Source Target\^\!* (TargetTerm TQ\^\!*)}
\item \textbf{by} (\textit{simp add: STCal-steps})
\item \textbf{ultimately obtain} \textit{R\^\!* where} \textit{B3: TargetTerm TR \rightarrow (STCal Source Target\^\!* R\^\!*}
\item \textbf{and} \textit{B4: (R\^\!, TargetTerm TQ\^\!) \in Rel}
\item \textbf{using} \textit{conSim}
\item \textbf{by} \textit{blast}
\item from \textit{B3 obtain} \textit{TR\^\!* where} \textit{B5: TR\^\!* \in T R\^\!* and B6: TR \rightarrow Target\^\!* TR\^\!*}
\item \textbf{by} (\textit{auto simp add: STCal-steps})
\item from \textit{B4 B5 trel have} \textit{(TR\^\!, TQ\^\!) \in TRel\^\!*}
\item \textbf{by} \textit{simp}
\item from \textit{this B2 have} \textit{(TR\^\!, TP\^\!) \in TRel\^\!*}
\item \textbf{by} \textit{simp}
\item with \textit{B6 show} \exists \textit{TR\^\!.\ TR \rightarrow Target\^\!* TR\^\! \land (TR\^\!, TP\^\!) \in TRel\^\!*}
\item \textbf{by} \textit{blast}
\item \textbf{qed}
\item \textbf{qed}

lemma (in encoding) \textit{indRelTEQ-impl-TRel-is-weak-reduction-contrasimulation}:

\begin{itemize}
\item \textbf{fixes} \textit{TRel} :: \textit{\'(procT \times 'procT\') set}
\item \textbf{assumes} \textit{conSim}:: weak-reduction-contrasimulation (\textit{indRelTEQ TRel}) (\textit{STCal Source Target})
\item \textbf{shows} weak-reduction-contrasimulation (\textit{TRel\^\!*}) \textit{Target}
\item using \textit{conSim} \textit{indRelTEQ-target[\textbf{where} TRel=\textit{TRel}] indRelTEQ-to-\textit{TRel}(.)[\textbf{where} TRel=\textit{TRel}] trans-closure-of-\textit{TRel-refl-cond}
\item rel-with-target-impl-reflC-transC-\textit{TRel-is-weak-reduction-contrasimulation}[\textbf{where} Rel=\textit{indRelTEQ TRel and TRel=\textit{TRel}]}
\item \textbf{by} \textit{blast}
\end{itemize}
by blast

lemma (in encoding-wrt-barbs) indRelRTPO-impl-TRel-is-weak-barbed-contrasimulation:
fixes TRel :: ('procT × 'procT) set
assumes conSim: weak-barbed-contrasimulation (indRelRTPO TRel) (STCalWB SWB TWB)
shows weak-barbed-contrasimulation (TRel⁺) TWB
proof
  from conSim show weak-reduction-contrasimulation (TRel⁺) (Calculus TWB)
    using indRelRTPO-impl-TRel-is-weak-reduction-contrasimulation[where TRel=TRel]
    by (simp add: STCalWB-def calS calT)
next
  from conSim show rel-weakly-preserves-barbs (TRel⁺) TWB
    using indRelRTPO-impl-TRel-weakly-preserves-barbs[where TRel=TRel]
    weak-preservation-of-barbs-and-closures(2)[where Rel=TRel and CWB=TWB]
    by blast
qed

lemma (in encoding-wrt-barbs) indRelLTPO-impl-TRel-is-weak-barbed-contrasimulation:
fixes TRel :: ('procT × 'procT) set
assumes conSim: weak-barbed-contrasimulation (indRelLTPO TRel) (STCalWB SWB TWB)
shows weak-barbed-contrasimulation (TRel⁺) TWB
proof
  from conSim show weak-reduction-contrasimulation (TRel⁺) (Calculus TWB)
    using indRelLTPO-impl-TRel-is-weak-reduction-contrasimulation[where TRel=TRel]
    by (simp add: STCalWB-def calS calT)
next
  from conSim show rel-weakly-preserves-barbs (TRel⁺) TWB
    using indRelLTPO-impl-TRel-weakly-preserves-barbs[where TRel=TRel]
    weak-preservation-of-barbs-and-closures(2)[where Rel=TRel and CWB=TWB]
    by blast
qed

lemma (in encoding-wrt-barbs) indRelTEQ-impl-TRel-is-weak-barbed-contrasimulation:
fixes TRel :: ('procT × 'procT) set
assumes conSim: weak-barbed-contrasimulation (indRelTEQ TRel) (STCalWB SWB TWB)
shows weak-barbed-contrasimulation (TRel∗) TWB
proof
  from conSim show weak-reduction-contrasimulation (TRel∗) (Calculus TWB)
    using indRelTEQ-impl-TRel-is-weak-reduction-contrasimulation[where TRel=TRel]
    by (simp add: STCalWB-def calS calT)
next
  from conSim show rel-weakly-preserves-barbs (TRel∗) TWB
    using indRelTEQ-impl-TRel-weakly-preserves-barbs[where TRel=TRel]
    weak-preservation-of-barbs-and-closures(3)[where Rel=TRel and CWB=TWB]
    by blast
qed

If indRelRTPO, indRelLTPO, or indRelTEQ is a coupled simulation then so is the corresponding
target term relation.

lemma (in encoding) indRelRTPO-impl-TRel-is-weak-reduction-coupled-simulation:
fixes TRel :: ('procT × 'procT) set
assumes couSim: weak-reduction-coupled-simulation (indRelRTPO TRel) (STCal Source Target)
shows weak-reduction-coupled-simulation (TRel⁺) Target
proof
  from couSim show weak-reduction-contrasimulation (TRel⁺) (Calculus TWB)
    using indRelRTPO-impl-TRel-is-weak-reduction-contrasimulation[where TRel=TRel]
    by (simp add: STCalWB-def calS calT)
next
  from couSim show rel-weakly-preserves-barbs (TRel⁺) TWB
    using indRelRTPO-impl-TRel-weakly-preserves-barbs[where TRel=TRel]
    weak-preservation-of-barbs-and-closures(2)[where Rel=TRel and CWB=TWB]
    by blast
lemma (in encoding) indRelLTPO-impl-TRel-is-weak-reduction-coupled-simulation:
fixes TRel :: ('procT × 'procT) set

\textbf{assumes} \textit{couSim}: weak-reduction-coupled-simulation (\textit{indRelLTPO} \textit{TRel}) (\textit{STCal Source Target})
\textbf{shows} weak-reduction-coupled-simulation (\textit{TRel}⁺) \textit{Target}
\textbf{using} \textit{couSim} weak-reduction-coupled-simulation-versus-simulation-and-contrasimulation
\textit{refl \textit{indRelLTPO-impl-TRel-is-weak-reduction-contrasimulation}[where \textit{TRel=\textit{TRel}]}\textit{indRelLTPO-impl-TRel-is-weak-reduction-contrasimulation}[where \textit{TRel=\textit{TRel}]}\textit{by blast}
\textnormal{\textbf{by blast}}
\textbf{lemma (in encoding) \textit{indRelTEQ-impl-TRel-is-weak-reduction-coupled-simulation}}:
\textbf{fixes} \textit{TRel} :: ('procT × 'procT) set
\textbf{assumes} \textit{couSim}: weak-reduction-coupled-simulation (\textit{indRelTEQ} \textit{TRel}) (\textit{STCal Source Target})
\textbf{shows} weak-reduction-coupled-simulation (\textit{TRel}⁺) \textit{Target}
\textbf{using} \textit{couSim} weak-reduction-coupled-simulation-versus-simulation-and-contrasimulation
\textit{refl \textit{indRelTEQ-impl-TRel-is-weak-reduction-contrasimulation}[where \textit{TRel=\textit{TRel}]}\textit{indRelTEQ-impl-TRel-is-weak-reduction-contrasimulation}[where \textit{TRel=\textit{TRel}]}\textit{by blast}
\textnormal{\textbf{by blast}}
\textbf{lemma (in encoding-wrt-barbs) \textit{indRelRTPO-impl-TRel-is-weak-barbed-coupled-simulation}}:
\textbf{fixes} \textit{TRel} :: ('procT × 'procT) set
\textbf{assumes} \textit{couSim}: weak-barbed-coupled-simulation (\textit{indRelRTPO} \textit{TRel}) (\textit{STCalWB SWB TWB})
\textbf{shows} weak-barbed-coupled-simulation (\textit{TRel}⁺) \textit{TWB}
\textbf{using} \textit{couSim} weak-barbed-coupled-simulation-versus-simulation-and-contrasimulation
\textit{refl \textit{indRelRTPO-impl-TRel-is-weak-barbed-contrasimulation}[where \textit{TRel=\textit{TRel}]}\textit{indRelRTPO-impl-TRel-is-weak-barbed-contrasimulation}[where \textit{TRel=\textit{TRel}]}\textit{by blast}
\textnormal{\textbf{by blast}}
\textbf{lemma (in encoding-wrt-barbs) \textit{indRelLTPO-impl-TRel-is-weak-barbed-coupled-simulation}}:
\textbf{fixes} \textit{TRel} :: ('procT × 'procT) set
\textbf{assumes} \textit{couSim}: weak-barbed-coupled-simulation (\textit{indRelLTPO} \textit{TRel}) (\textit{STCalWB SWB TWB})
\textbf{shows} weak-barbed-coupled-simulation (\textit{TRel}⁺) \textit{TWB}
\textbf{using} \textit{couSim} weak-barbed-coupled-simulation-versus-simulation-and-contrasimulation
\textit{refl \textit{indRelLTPO-impl-TRel-is-weak-barbed-contrasimulation}[where \textit{TRel=\textit{TRel}]}\textit{indRelLTPO-impl-TRel-is-weak-barbed-contrasimulation}[where \textit{TRel=\textit{TRel}]}\textit{by blast}
\textnormal{\textbf{by blast}}
\textbf{lemma (in encoding-wrt-barbs) \textit{indRelTEQ-impl-TRel-is-weak-barbed-coupled-simulation}}:
\textbf{fixes} \textit{TRel} :: ('procT × 'procT) set
\textbf{assumes} \textit{couSim}: weak-barbed-coupled-simulation (\textit{indRelTEQ} \textit{TRel}) (\textit{STCalWB SWB TWB})
\textbf{shows} weak-barbed-coupled-simulation (\textit{TRel}⁺) \textit{TWB}
\textbf{using} \textit{couSim} weak-barbed-coupled-simulation-versus-simulation-and-contrasimulation
\textit{refl \textit{indRelTEQ-impl-TRel-is-weak-barbed-contrasimulation}[where \textit{TRel=\textit{TRel}]}\textit{indRelTEQ-impl-TRel-is-weak-barbed-contrasimulation}[where \textit{TRel=\textit{TRel}]}\textit{by blast}
\textnormal{\textbf{by blast}}
\textbf{If \textit{indRelRTPO}, \textit{indRelLTPO}, or \textit{indRelTEQ} is a correspondence simulation then so is the corresponding target term relation.}
\textbf{lemma (in encoding) \textit{rel-with-target-impl-transC-TRel-is-weak-reduction-correspondence-simulation}}:
\textbf{fixes} \textit{TRel} :: ('procT × 'procT) set
\textbf{and} \textit{Rel} :: ('procS, 'procT) Proc × ('procS, 'procT) Proc set
\textbf{assumes} \textit{corSim}: weak-reduction-correspondence-simulation \textit{Rel} (\textit{STCal Source Target})
\textbf{and} \textit{target}: ∀ \textit{T1} \textit{T2}. (\textit{T1}, \textit{T2}) ∈ \textit{TRel} → (\textit{TargetTerm T1}, \textit{TargetTerm T2}) ∈ \textit{Rel}
\textbf{and} \textit{trel}: ∀ \textit{T1} \textit{T2}. (\textit{TargetTerm T1}, \textit{TargetTerm T2}) ∈ \textit{Rel} → (\textit{T1}, \textit{T2}) ∈ \textit{TRel}⁺
\textbf{shows} weak-reduction-correspondence-simulation (\textit{TRel}⁺) \textit{Target}
\textbf{proof}\textnormal{–}
\textbf{from} \textit{corSim} \textit{target} \textit{trel} \textit{have} \textit{A}: weak-reduction-simulation (\textit{TRel}⁺) \textit{Target}
\textbf{using} \textit{rel-with-target-impl-transC-TRel-is-weak-reduction-simulation}[where \textit{TRel=\textit{TRel}}]
\textbf{and} \textit{Rel=\textit{Rel}}
\textbf{by blast}
\textbf{moreover have} ∀ \textit{P} \textit{Q} \textit{Q'}. (\textit{P}, \textit{Q}) ∈ \textit{TRel}⁺ ∧ \textit{Q} ℮\textit{Target* Q'}
\textbf{proof clarify}

110
fix $TP \ TQ \ TQ'$
assume $(TP, TQ) \in TRel$ and $TQ \rightarrow Target* TQ'$
thus $\exists TP'' TQ'' . TP \rightarrow Target* TP'' \land TQ' \rightarrow Target* TQ'' \land (TP'', TQ'') \in TRel$
proof (induct arbitrary: $TQ'$)
  fix $TQ TQ'$
  assume $(TP, TQ) \in TRel$
  with target have $(TargetTerm TP, TargetTerm TQ) \in Rel$
    by blast
  moreover assume $TQ \rightarrow Target* TQ'$
  hence $TargetTerm TQ \rightarrow(STCal Source Target)* (TargetTerm TQ')$
    by (simp add: STCal-steps)
  ultimately obtain $P'' Q''$ where $A2: TargetTerm TP \rightarrow(STCal Source Target)* P''$
    and $A3: TargetTerm TQ' \rightarrow(STCal Source Target)* Q''$ and $A4: (P'', Q'') \in Rel$
    using corSim
    by blast
  from $A2$ obtain $TP''$ where $A5: TP \rightarrow Target* TP''$ and $A6: TP'' \in T P''$
    by (auto simp add: STCal-steps)
  from $A3$ obtain $TQ''$ where $A7: TQ' \rightarrow Target* TQ''$ and $A8: TQ'' \in T Q''$
    by (auto simp add: STCal-steps)
  from $A4 \ A6 \ A8$ trel have $(TP'', TQ'') \in TRel$
    by blast
  with $A5 \ A7$
  show $\exists TP'' TQ'' . TP \rightarrow Target* TP'' \land TQ' \rightarrow Target* TQ'' \land (TP'', TQ'') \in TRel$
    by blast
next
  case (step $TQ TR TR'$)
  assume $\bigwedge TQ'. TQ \rightarrow Target* TQ' \Rightarrow \exists TP'' TQ''. TP \rightarrow Target* TP'' \land TQ' \rightarrow Target* TQ'' \land (TP'', TQ'') \in TRel$
  moreover assume $(TQ, TR) \in TRel$
  hence $\bigwedge TR'. TR \rightarrow Target* TR'$
    $\rightarrow \exists TQ'' TR''. TQ \rightarrow Target* TQ'' \land TR' \rightarrow Target* TR'' \land (TQ'', TR'') \in TRel$
  proof clarify
    fix $TR'$
    assume $(TQ, TR) \in TRel$
    with target have $(TargetTerm TQ, TargetTerm TR) \in Rel$
      by simp
    moreover assume $TR \rightarrow Target* TR'$
    hence $TargetTerm TR \rightarrow(STCal Source Target)* (TargetTerm TR')$
      by (simp add: STCal-steps)
    ultimately obtain $Q'' R''$ where $B1: TargetTerm TQ \rightarrow(STCal Source Target)* Q''$
      and $B2: TargetTerm TR' \rightarrow(STCal Source Target)* R''$ and $B3: (Q'', R'') \in Rel$
      using corSim
      by blast
    from $B1$ obtain $TQ''$ where $B4: TQ'' \in T Q''$ and $B5: TQ \rightarrow Target* TQ''$
      by (auto simp add: STCal-steps)
    from $B2$ obtain $TR''$ where $B6: TR'' \in T R''$ and $B7: TR' \rightarrow Target* TR''$
      by (auto simp add: STCal-steps)
    from $B3 \ B4 \ B6$ trel have $(TQ'', TR'') \in TRel$
      by simp
    with $B5 \ B7$
    show $\exists TQ'' TR''. TQ \rightarrow Target* TQ'' \land TR' \rightarrow Target* TR'' \land (TQ'', TR'') \in TRel$
      by blast
  qed
  moreover have trans $(TRel^+)$
    by simp
  moreover assume $TR \rightarrow Target* TR'$
  ultimately
  show $\exists TP'' TR''. TP \rightarrow Target* TP'' \land TR' \rightarrow Target* TR'' \land (TP'', TR'') \in TRel$
    using $A$ reduction-correspondence-simulation-condition-trans[where Rel=TRel$^+$
      and $Cal=Target$]
    by blast
qed

lemma (in encoding) \textit{indRelRTPO-impl-TRel-is-weak-reduction-correspondence-simulation}:

\begin{itemize}
\item \textbf{fixes} \( TRel :: ('procT \times 'procT) \) set
\item \textbf{assumes} \( \textit{cSim} :: \textit{weak-reduction-correspondence-simulation} \) (\( \textit{indRelRTPO} \textit{TRel} \) \( (\textit{STCal Source Target}) \)
\item \textbf{shows} \( \textit{cSim indRelRTPO.\target[\where TRel=TRel]} \textit{indRelRTPO-to-TRel(\)\where TRel=TRel}\)
\item \textbf{using} \( \textit{cSim indRelRTPO.\target[\where TRel=TRel]} \textit{indRelRTPO-to-TRel(\)\where TRel=TRel}\)
\end{itemize}

\textit{by blast}

\begin{itemize}
\item \textbf{lemma (in encoding) \textit{indRelLTP0-impl-TRel-is-weak-reduction-correspondence-simulation}:
\item \textbf{fixes} \( TRel :: ('procT \times 'procT) \) set
\item \textbf{assumes} \( \textit{cSim} :: \textit{weak-reduction-correspondence-simulation} \) (\( \textit{indRelLTO} \textit{TRel} \) \( (\textit{STCal Source Target}) \)
\item \textbf{shows} \( \textit{cSim indRelLTO.\target[\where TRel=TRel]} \textit{indRelLTO-to-TRel(\)\where TRel=TRel}\)
\item \textbf{using} \( \textit{cSim indRelLTO.\target[\where TRel=TRel]} \textit{indRelLTO-to-TRel(\)\where TRel=TRel}\)
\end{itemize}

\textit{by blast}

\begin{itemize}
\item \textbf{lemma (in encoding) \textit{rel-with-target-impl-refC-transC-TRel-is-weak-reduction-correspondence-simulation}:
\item \textbf{fixes} \( TRel :: ('procT \times 'procT) \) set
\item \textbf{and} \( \textit{Rel :: ('procS \times 'procT) \text{Proc \times ('procS \times 'procT) \text{Proc set}}\)
\item \textbf{assumes} \( \textit{corSim :: weak-reduction-correspondence-simulation Rel (\textit{STCal Source Target})}\)
\item \textbf{and} \( \textit{trel :: \forall T1 T2 \ (T1, T2) \in TRel \rightarrow \textit{TargetTerm T1, TargetTerm T2) \in Rel}\)
\item \textbf{and} \( \textit{trel :: \forall T1 T2 \ (T1, T2) \in TRel^* \rightarrow \textit{TargetTerm T1, TargetTerm T2) \in Rel \rightarrow (T1, T2) \in TRel^*}\)
\item \textbf{shows} \( \textit{weak-reduction-correspondence-simulation (TRel^*) Target}\)
\item \textbf{proof} –
\item \textbf{from corSim target trel have A :: \textit{weak-reduction-simulation (TRel^*) Target\)
\item \textbf{using} \( \textit{rel-with-target-impl-refC-transC-TRel-is-weak-reduction-correspondence-simulation[\where TRel=TRel}\)
\item \textbf{and} \( \textit{Rel=Rel}\)
\textit{by blast}
\end{itemize}

\begin{itemize}
\item \textbf{moreover have} \( \forall P Q Q'. (P, Q) \in TRel^* \land Q \rightarrow \textit{Target}* Q' \rightarrow (\exists P'' Q'\cdot P \rightarrow \textit{Target}* P'' \land Q' \rightarrow \textit{Target}* Q'' \land (P'', Q') \in TRel^*)\)
\item \textbf{proof} \textit{clarify}
\item \textbf{fix} \( TP TQ TQ'\)
\item \textbf{assume} \( (TP, TQ) \in TRel^* \land TQ \rightarrow \textit{Target}* TQ'\)
\item \textbf{thus} \( \exists P'' TQ''\cdot TP \rightarrow \textit{Target}* P'' \land TQ' \rightarrow \textit{Target}* TQ'' \land (TQ'', TQ') \in TRel^*\)
\item \textbf{proof} \textit{(induct arbitrary: TQ')}
\item \textbf{fix} \( TQ'\)
\item \textbf{assume} \( TP \rightarrow \textit{Target}* TQ'\)
\item \textbf{moreover have} \( TQ' \rightarrow \textit{Target}* TQ'\)
\item \textbf{by} \( \textit{(simp add: steps-refl)}\)
\item \textbf{moreover have} \( (TQ', TQ') \in TRel^*\)
\item \textbf{by} \( \textit{simp}\)
\item \textbf{ultimately show} \( \exists TQ'' TQ''\cdot TP \rightarrow \textit{Target}* TQ'' \land TQ' \rightarrow \textit{Target}* TQ'' \land (TQ'', TQ') \in TRel^*\)
\item \textbf{by} \( \textit{blast}\)
\item \textbf{next}
\item \textbf{case} \( \textit{(step TQ TR TR')}\)
\item \textbf{assume} \( \land TQ'. TQ \rightarrow \textit{Target}* TQ' \rightarrow \exists TQ'' TQ''. TP \rightarrow \textit{Target}* TQ'' \land TQ' \rightarrow \textit{Target}* TQ'' \land (TQ'', TQ') \in TRel^*\)
\item \textbf{moreover assume} \( (TQ, TR) \in TRel\)
\item \textbf{with} \( \textit{corSim have} \land TR'. TR \rightarrow \textit{Target}* TR' \rightarrow \exists TQ'' TQ''. TQ \rightarrow \textit{Target}* TQ'' \land TR' \rightarrow \textit{Target}* TQ'' \land (TQ'', TQ') \in TRel^*\)
\item \textbf{proof} \textit{clarify}
\item \textbf{fix} \( TR'\)
\end{itemize}
assume \((TQ, TR) \in TRel\)

with target have \((\text{TargetTerm } TQ, \text{TargetTerm } TR) \in \text{Rel}\)

by simp

moreover assume \(TR \rightarrow \text{Target* } TR'\)

hence \(\text{TargetTerm } TR \rightarrow (\text{STCal Source Target})* \ (\text{TargetTerm } TR')\)

by (simp add: \text{STCal-steps})

ultimately obtain \(Q'' R''\) where \(B1: \text{TargetTerm } TQ \rightarrow (\text{STCal Source Target})* Q''\)

and \(B2: \text{TargetTerm } TR' \rightarrow (\text{STCal Source Target})* R''\) and \(B3: (Q'', R'') \in \text{Rel}\)

using \(\text{corSim}\)

by blast

from \(B1\) obtain \(TQ''\) where \(B4: TQ'' \in T Q''\) and \(B5: TQ \rightarrow \text{Target* } TQ''\)

by (auto simp add: \text{STCal-steps})

from \(B2\) obtain \(TR''\) where \(B6: TR'' \in T R''\) and \(B7: TR' \rightarrow \text{Target* } TR''\)

by (auto simp add: \text{STCal-steps})

from \(B3 B4 B6\) trel have \((TQ'', TR'') \in TRel\)

by simp

with \(B5 B7\)

show \(\exists TQ'' TR''. TQ \rightarrow \text{Target* } TQ'' \land TR' \rightarrow \text{Target* } TR'' \land (TQ'', TR'') \in TRel\)

by blast

qed

moreover assume \(TR \rightarrow \text{Target* } TR'\)

moreover have \(\text{trans } (\text{TRel}^*)\)

using \(\text{trans-trancl[of TRel]}\)

by simp

ultimately show \(\exists TP'' TR''. TP \rightarrow \text{Target* } TP'' \land TR' \rightarrow \text{Target* } TR'' \land (TP'', TR'') \in TRel\)

using \(A\) \text{ reduction-correspondence-simulation-condition-trans[where Rel=TRel*}

and \(\text{Cal=Target]}\)

by blast

qed

lemma (in encoding) \(\text{indRelTEQ-impl-TRel-is-weak-reduction-correspondence-simulation}:\)

fixes \(\text{TRel} :: (\text{'procT} \times \text{'procT})\) set

assumes \(\text{corSim}: \text{weak-reduction-correspondence-simulation } (\text{indRelTEQ TRel}) \ (\text{STCal Source Target)}\)

shows \(\text{weak-reduction-correspondence-simulation } (\text{TRel}^*) \ (\text{Target)}\)

using \(\text{corSim indRelTEQ.target[where TRel=TRel]} \text{indRelTEQ-to-TRel}\(4)[where TRel=TRel]

\text{trans-closure-of-TRel-refl-cond}

\text{rel-with-target-impl-reflC-transC-TRel-is-weak-reduction-correspondence-simulation[}

\text{where Rel=indRelTEQ TRel and TRel=TRel]}\)

by blast

lemma (in encoding-wrt-barbs) \(\text{indRelRTPO-impl-TRel-is-weak-barbed-correspondence-simulation}:\)

fixes \(\text{TRel} :: (\text{'procC} \times \text{'procT})\) set

assumes \(\text{corSim}: \text{weak-barbed-correspondence-simulation } (\text{indRelRTPO TRel}) \ (\text{STCalWB SWB TWB)}\)

shows \(\text{weak-barbed-correspondence-simulation } (\text{TRel}^*) \ (\text{TWB)}\)

proof

from \(\text{corSim}\) show \(\text{weak-reduction-correspondence-simulation } (\text{TRel}^*) \ (\text{Calculus TWB)}\)

using \(\text{indRelRTPO-impl-TRel-is-weak-reduction-correspondence-simulation[where TRel=TRel]}

\text{trans-closure-of-TRel-refl-cond}

\text{rel-with-target-impl-reflC-transC-TRel-is-weak-reduction-correspondence-simulation[}

\text{where Rel=indRelTEQ TRel and TRel=TRel]}\)

by simp add: \text{STCalWB-def calS calT)}

next

from \(\text{corSim}\) show \(\text{rel-weakly-respects-barbs } (\text{TRel}^*) \ (\text{TWB)}\)

using \(\text{indRelRTPO-impl-TRel-weakly-respects-barbs[where TRel=TRel]}

\text{trans-closure-of-TRel-refl-cond}

\text{rel-with-target-impl-reflC-transC-TRel-is-weak-reduction-correspondence-simulation[}

\text{where Rel=indRelTEQ TRel and TRel=TRel]}\)

by blast

qed

lemma (in encoding-wrt-barbs) \(\text{indRelLTPO-impl-TRel-is-weak-barbed-correspondence-simulation}:\)

fixes \(\text{TRel} :: (\text{'procC} \times \text{'procT})\) set
assumes corSim: weak-barbed-correspondence-simulation (indRelLTPO TRel) (STCalWB SWB TWB)
shows weak-barbed-correspondence-simulation (TRel⁺) TWB
proof
next
from corSim show weak-reduction-correspondence-simulation (TRel⁺) (Calculus TWB)
  using indRelLTPO-impl-TRel-is-weak-reduction-correspondence-simulation[where TRel= TRel]
  by (simp add: STCalWB-def calS calT)
qed

lemma (in encoding-wrt-barbs) indRelTEQ-impl-TRel-is-weak-barbed-correspondence-simulation:
  fixes TRel :: ('procT × 'procT) set
  assumes corSim: weak-barbed-correspondence-simulation (indRelTEQ TRel) (STCalWB SWB TWB)
  shows weak-barbed-correspondence-simulation (TRel⁺) TWB
proof
  from corSim show rel-weakly-respects-barbs (TRel⁺) TWB
    using indRelTEQ-impl-TRel-weakly-respects-barbs[where TRel= TRel]
    by blast
  next
  from corSim show rel-weakly-respects-barbs (TRel⁺) TWB
    using indRelTEQ-impl-TRel-weakly-respects-barbs[where TRel= TRel]
    by blast
  qed

If indRelRTPO, indRelLTPO, or indRelTEQ is a bisimulation then so is the corresponding target term relation.

lemma (in encoding) rel-with-target-impl-transC-TRel-is-weak-reduction-bisimulation:
  fixes TRel :: ('procT × 'procT) set
  and Rel :: ('procS, 'procT) Proc × ('procS, 'procT) Proc set
  assumes bisim: weak-reduction-bisimulation Rel (STCal Source Target)
    and target: ∀ T1 T2. (T1, T2) ∈ TRel ⟹ (TargetTerm T1, TargetTerm T2) ∈ Rel
    and trel: ∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel ⟹ (T1, T2) ∈ TRel⁺
  shows weak-reduction-bisimulation (TRel⁺⁺) Target
proof
  from bisim target trel show weak-reduction-simulation (TRel⁺) Target
    using rel-with-target-impl-transC-TRel-is-weak-reduction-simulation[where TRel= TRel
      and Rel= Rel]
    by blast
next
  show ∀ P Q Q’. (P, Q) ∈ TRel⁺⁺ ∧ Q ⟹ Target* Q’ ⟹ (∃ P’. P ⟹ Target* P’ ∧ (P’, Q’) ∈ TRel⁺⁺)
proof clarify
  fix TP TQ TQ’
  assume (TP, TQ) ∈ TRel⁺⁺ and TQ ⟹ Target* TQ’
  thus ∃ TP’. TP ⟹ Target* TP’ ∧ (TP’, TQ’) ∈ TRel⁺⁺
proof (induct arbitrary: TQ’)
  fix TQ TQ’
  assume (TP, TQ) ∈ TRel
  with target have (TargetTerm TP, TargetTerm TQ) ∈ Rel
  by simp
  moreover assume TQ ⟹ Target* TQ’
  hence TargetTerm TQ ⟹ (STCal Source Target)* (TargetTerm TQ’)
  by (simp add: STCal-steps)
  ultimately obtain P’ where A2: TargetTerm TP ⟹ (STCal Source Target)* P’
    and A3: (P’, TargetTerm TQ’) ∈ Rel
    using bisim
    by blast

114
from $A2$ obtain $TP'$ where $A4$: $TP \longrightarrow \text{Target* } TP'$ and $A5$: $TP' \in T' P'$
by (auto simp add: STCal-steps)
from $A3$ $A5$ trel have $(TP', TQ') \in TRel$^+
by simp
with $A4$ show $\exists TP'. TP \longrightarrow \text{Target* } TP' \land (TP', TQ') \in TRel$^+
by blast

next

  case (step $TQ \ TR \ TR'$)
  assume $(TQ, TR) \in TRel$
  with target have $(\text{TargetTerm } TQ, \text{TargetTerm } TR) \in \text{Rel}$
  by simp
  moreover assume $TR \longrightarrow \text{Target* } TR'$
  hence $\text{TargetTerm } TR \longrightarrow (\text{STCal Source Target})^* (\text{TargetTerm } TR')$
  by (simp add: STCal-steps)
  ultimately obtain $Q'$ where $B1$: $\text{TargetTerm } TQ \longrightarrow (\text{STCal Source Target})^* Q'$
  and $B2$: $(Q', \text{TargetTerm } TR') \in \text{Rel}$
  using bisim
  by blast
from $B1$ obtain $TQ'$ where $B3$: $TQ' \in T' Q'$ and $B4$: $TQ \longrightarrow \text{Target* } TQ'$
by (auto simp add: STCal-steps)
assume $\bigwedge TQ'. TQ \longrightarrow \text{Target* } TQ' \implies \exists TP'. TP \longrightarrow \text{Target* } TP' \land (TP', TQ') \in TRel$^+
with $B4$ obtain $TP'$ where $B5$: $TP \longrightarrow \text{Target* } TP'$ and $B6$: $(TP', TQ') \in TRel$^+
by blast
from $B2$ $B3$ trel have $(TQ', TR') \in TRel$^+
by simp
with $B6$ have $(TP', TR') \in TRel$^+
by simp
with $B5$ show $\exists TP'. TP \longrightarrow \text{Target* } TP' \land (TP', TR') \in TRel$^+
by blast
qed
qed

lemma (in encoding) indRelRTPO-impl-TRel-is-weak-reduction-bisimulation:
  fixes $TRel :: (\text{'procT} \times \text{'procT})$ set
  assumes bisim: weak-reduction-bisimulation $(\text{indRelRTPO TRel})$ (STCal Source Target)
  shows weak-reduction-bisimulation $(TRel^+)$ Target
  using bisim indRelRTPO.target[where TRel=TRel] indRelRTPO-to-TRel(4)[where TRel=TRel]
  rel-with-target-impl-transC-TRel-is-weak-reduction-bisimulation[where Rel=indRelRTPO TRel and TRel=TRel]
  by blast

lemma (in encoding) indRelLTPO-impl-TRel-is-weak-reduction-bisimulation:
  fixes $TRel :: (\text{'procT} \times \text{'procT})$ set
  assumes bisim: weak-reduction-bisimulation $(\text{indRelLTPO TRel})$ (STCal Source Target)
  shows weak-reduction-bisimulation $(TRel^+)$ Target
  using bisim indRelLTPO.target[where TRel=TRel] indRelLTPO-to-TRel(4)[where TRel=TRel]
  rel-with-target-impl-transC-TRel-is-weak-reduction-bisimulation[where Rel=indRelLTPO TRel and TRel=TRel]
  by blast

lemma (in encoding) rel-with-target-impl-reflC-transC-TRel-is-weak-reduction-bisimulation:
  fixes $TRel :: (\text{'procT} \times \text{'procT})$ set
  and $\text{Rel} :: (\text{'procS}, \text{'procT})$ $\text{Proc} \times (\text{'procS}, \text{'procT})$ $\text{Proc}$ set
  assumes bisim: weak-reduction-bisimulation $\text{Rel}$ (STCal Source Target)
  and trel: $\forall T1 T2. (T1, T2) \in TRel \longrightarrow (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}$
  and trel: $\forall T1 T2. (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel} \longrightarrow (T1, T2) \in TRel^+$
  shows weak-reduction-bisimulation $(TRel^+)$ Target
proof
  from bisim target trel show weak-reduction-simulation $(TRel^+)$ Target
  using rel-with-target-impl-reflC-transC-TRel-is-weak-reduction-simulation[where TRel=TRel]
\begin{quote}
and \( Rel = Rel \)

by blast

next

show \( \forall P \ Q \ Q'. (P, \ Q) \in TRel^* \land Q \rightarrow TTarget \ast \ Q' \rightarrow (\exists P'. P \rightarrow TTarget \ast \ P' \land (P', \ Q') \in TRel^*) \)

proof clarify

fix \( TP \ TQ \ TQ' \)

assume \( (TP, \ TQ) \in TRel^* \) and \( TQ \rightarrow TTarget \ast \ TQ' \)

thus \( \exists TP'. TP \rightarrow TTarget \ast TP' \land (TP', \ TQ') \in TRel^* \)

proof (induct arbitrary: \( TQ' \))

fix \( TQ' \)

assume \( TP \rightarrow TTarget \ast \ TQ' \)

moreover have \( (TQ', \ TQ') \in TRel^* \)

by simp

ultimately show \( \exists TP'. TP \rightarrow TTarget \ast TP' \land (TP', \ TQ') \in TRel^* \)

by blast

next

case \( (\text{step} \ TQ \ TR \ TR') \)

assume \( (TQ, \ TR) \in TRel \)

with \( \text{target have} \ (TTargetTerm \ TQ, \ TargetTerm \ TR) \in Rel \)

by simp

moreover assume \( TR \rightarrow TTarget \ast \ TR' \)

hence \( TargetTerm \ TR \rightarrow (\text{STCal Source Target}) \ast (TargetTerm \ TR') \)

by (simp add: STCal-steps)

ultimately obtain \( Q' \) where \( B1: \text{TargetTerm \ TQ} \rightarrow (\text{STCal Source Target}) \ast Q' \)

and \( B2: \text{TargetTerm \ TR'} \in Rel \)

using bisim

by blast

from \( B1 \) obtain \( TQ' \) where \( B3: \ TQ' \in T \ Q' \) and \( B4: \ TP \rightarrow TTarget \ast \ TQ' \)

by (auto simp add: STCal-steps)

assume \( (\forall TQ'. \ TQ \rightarrow TTarget \ast TQ' \rightarrow \exists TP'. TP \rightarrow TTarget \ast TP' \land (TP', \ TQ') \in TRel^* \)

with \( B4 \) obtain \( TP'' \) where \( B5: \ TP \rightarrow TTarget \ast TP' \) and \( B6: (TP', \ TQ') \in TRel^* \)

by blast

from \( B2 \ B3 \ trel \) have \( (TQ', \ TR') \in TRel^* \)

by simp

with \( B6 \) have \( (TP', \ TR') \in TRel^* \)

by simp

with \( B5 \) show \( \exists TP'. TP \rightarrow TTarget \ast TP' \land (TP', \ TR') \in TRel^* \)

by blast

qed

qed

lemma (in encoding) \( \text{indRelTEQ-impl-TRel-is-weak-reduction-bisimulation} \):

\begin{itemize}
\item \( TRel \ :: \ ('\text{procT} \times \ '\text{procT}) \ \text{set} \)
\item \( \text{assumes} \ \text{bisim: weak-reduction-bisimulation} \ (\text{indRelTEQ} \ TRel) \ (\text{STCal Source Target}) \)
\item \( \text{shows} \ \text{weak-reduction-bisimulation} \ (TRel^*) \ \text{Target} \)
\end{itemize}

using \( \text{bisim indRelTEQ, target [where TRel = TRel] \text{indRelTEQ-to-TRel} (\text{where TRel = TRel})} \)

\begin{itemize}
\item \( \text{trans-closure-of-TRel-refl-cond} \)
\item \( \text{rel-with-target-impl-reflC-transC-TRel-is-weak-reduction-bisimulation [where Rel = \text{indRelTEQ} \ TRel \ and TRel = TRel]} \)
\end{itemize}

by blast

lemma (in encoding) \( \text{rel-with-target-impl-transC-TRel-is-strong-reduction-bisimulation} \):

\begin{itemize}
\item \( TRel \ :: \ ('\text{procT} \times \ '\text{procT}) \ \text{set} \)
\item \( \text{and} \ \text{Rel} :: \ ('\text{procS}, \ '\text{procT}) \ \text{Proc} \times ('\text{procS}, \ '\text{procT}) \ \text{Proc} \ \text{set} \)
\item \( \text{assumes} \ \text{bisim: strong-reduction-bisimulation} \ (\text{Rel} \ (\text{STCal Source Target}) \) \)
\item \( \text{and} \ \text{target:} \ \forall T1 \ T2. (T1, \ T2) \in TRel \rightarrow (\text{TargetTerm \ T1, \ TargetTerm \ T2}) \in \text{Rel} \)
\item \( \text{and} \ \text{trel:} \ \forall T1 \ T2. (\text{TargetTerm \ T1, \ TargetTerm \ T2}) \in \text{Rel} \rightarrow (T1, \ T2) \in TRel^+ \)
\item \( \text{shows} \ \text{strong-reduction-bisimulation} \ (TRel^+ \) \ \text{Target} \)
\end{itemize}

proof

from \( \text{bisim target trel show strong-reduction-simulation} \ (TRel^+) \ \text{Target} \)
proof (induct arbitrary: TQ)
  fix TQ TQ'
  assume (TQ, TQ') ∈ TRel+ and TQ'⇒Target TQ'
  thus ∃TQ', TP'⇒Target TP' ∧ (TP', TQ') ∈ TRel+
  proof (auto simp add: STCal-step)
    blast
  qed
qed

lemma (in encoding) indRelRTPO-impl-TRel-is-strong-reduction-bisimulation:
  fixes TRel :: ('procT × 'procT) set
  assumes bisim: strong-reduction-bisimulation (indRelRTPO TRel) (STCal Source Target)
  shows strong-reduction-bisimulation (TRel+) Target
  using bisim indRelRTPO.target[where TRel=TRel] indRelRTPO-to-TRel( where TRel=TRel]
rel-with-target-impl-transC-TRel-is-strong-reduction-bisimulation[where Rel=indRelRTPO TRel and TRel=TRel]
lemma (in encoding) indRelLTPO-impl-TRel-is-strong-reduction-bisimulation:
  fixes TRel :: ('procT × 'procT) set
  assumes bisim: strong-reduction-bisimulation (indRelLTPO TRel) (STCal Source Target)
  shows strong-reduction-bisimulation (TRel+) Target
    using bisim indRelLTPO.to-TRel(4) indRelLTPO.trel-with-target-impl-transC-TRel-is-strong-reduction-bisimulation
   Rel=indRelLTPO TRel and TRel=TRel
  by blast

lemma (in encoding) rel-with-target-impl-reflC-transC-TRel-is-strong-reduction-bisimulation:
  fixes TRel :: ('procT × 'procT) set
  and Rel :: ('procS, 'procT) Proc × ('procS, 'procT) Proc set
  assumes bisim: strong-reduction-bisimulation Rel (STCal Source Target)
  and target: ∀ T1 T2. (T1, T2) ∈ TRel ➔ (TargetTerm T1, TargetTerm T2) ∈ Rel
  and trel: ∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel ➔ (T1, T2) ∈ TRel*
  shows strong-reduction-bisimulation (TRel*) Target
  proof
    from bisim target trel show strong-reduction-simulation (TRel*) Target
      using rel-with-target-impl-reflC-transC-TRel-is-strong-reduction-simulation[where Rel=Rel
       and TRel=TRel]
    by blast

next
  show ∀ P Q Q'. (P, Q) ∈ TRel* ∧ Q → Target Q' → (∃ P'. P → Target P' ∧ (P', Q') ∈ TRel*)
  proof clarify
    fix TP TQ TQ'
    assume (TP, TQ) ∈ TRel* and TQ → Target TQ'
    thus ∃ TP'. TP → Target TP' ∧ (TP', TQ') ∈ TRel*
    proof (induct arbitrary: TQ')
      fix TQ'
      assume TP → Target TQ'
      thus ∃ TP'. TP → Target TP' ∧ (TP', TQ') ∈ TRel*
      by blast
    next
      case (step TQ TR TR')
      assume (TQ, TR) ∈ TRel
      with target have (TargetTerm TQ, TargetTerm TR) ∈ Rel
      by simp
      moreover assume TR → Target TR'
      hence TargetTerm TR → (STCal Source Target) (TargetTerm TR')
      by (simp add: STCal-step)
      ultimately obtain Q' where B1: TargetTerm TQ → (STCal Source Target) Q'
       and B2: (Q', TargetTerm TR') ∈ Rel
          using bisim
          by blast
      from B1 obtain TQ' where B3: TQ' ∈ T Q' and B4: TQ' → Target TQ'
        by (auto simp add: STCal-step)
      assume TQ'. TQ → Target TQ' → ∃ TP'. TP → Target TP' ∧ (TP', TQ') ∈ TRel*
      with B4 obtain TP' where B5: TP → Target TP' and B6: (TP', TQ') ∈ TRel*
      by blast
      from B2 B3 trel have (TQ', TR') ∈ TRel*
        by simp
      with B6 have (TP', TR') ∈ TRel*
        by simp
      with B5 show ∃ TP'. TP → Target TP' ∧ (TP', TR') ∈ TRel*
        by blast
    qed
  qed
lemma (in encoding) indRelTEQ-impl-TRel-is-strong-reduction-bisimulation:
  fixes TRel :: ('procT × 'procT) set
  assumes bisim: strong-reduction-bisimulation (indRelTEQ TRel) (STCal Source Target)
  shows strong-reduction-bisimulation (TRel+) Target
    using bisim indRelTEQ.target[where TRel=TRel] indRelTEQ-to-TRel(4)[where TRel=TRel]
    trans-closure-of-TRel-refl-cond
  rel-with-target-impl-reflC-transC-TRel-is-strong-reduction-bisimulation[where
    Rel=indRelTEQ TRel and TRel=TRel]
  by blast

lemma (in encoding-wrt-barbs) indRelRTPO-impl-TRel-is-weak-barbed-bisimulation:
  fixes TRel :: ('procT × 'procT) set
  assumes bisim: weak-barbed-bisimulation (indRelRTPO TRel) (STCalWB SWB TBW)
  shows weak-barbed-bisimulation (TRel+) TBW
proof
  from bisim show weak-reduction-bisimulation (TRel+) (Calculus TBW)
    using indRelRTPO-impl-TRel-is-weak-reduction-bisimulation[where TRel=TRel]
    by (simp add: STCalWB-def calS calT)
next
  from bisim show rel-weakly-respects-barbs (TRel+) TBW
    using indRelRTPO-impl-TRel-weakly-respects-barbs[where TRel=TRel]
    weak-respection-of-barbs-and-closures(3)[where Rel=TRel and CWB=TBW]
    by blast
qed

lemma (in encoding-wrt-barbs) indRelLTPO-impl-TRel-is-weak-barbed-bisimulation:
  fixes TRel :: ('procT × 'procT) set
  assumes bisim: weak-barbed-bisimulation (indRelLTPO TRel) (STCalWB SWB TBW)
  shows weak-barbed-bisimulation (TRel+) TBW
proof
  from bisim show weak-reduction-bisimulation (TRel+) (Calculus TBW)
    using indRelLTPO-impl-TRel-is-weak-reduction-bisimulation[where TRel=TRel]
    by (simp add: STCalWB-def calS calT)
next
  from bisim show rel-weakly-respects-barbs (TRel+) TBW
    using indRelLTPO-impl-TRel-weakly-respects-barbs[where TRel=TRel]
    weak-respection-of-barbs-and-closures(3)[where Rel=TRel and CWB=TBW]
    by blast
qed

lemma (in encoding-wrt-barbs) indRelTEQ-impl-TRel-is-weak-barbed-bisimulation:
  fixes TRel :: ('procT × 'procT) set
  assumes bisim: weak-barbed-bisimulation (indRelTEQ TRel) (STCalWB SWB TBW)
  shows weak-barbed-bisimulation (TRel+) TBW
proof
  from bisim show weak-reduction-bisimulation (TRel+) (Calculus TBW)
    using indRelTEQ-impl-TRel-is-weak-reduction-bisimulation[where TRel=TRel]
    by (simp add: STCalWB-def calS calT)
next
  from bisim show rel-weakly-respects-barbs (TRel+) TBW
    using indRelTEQ-impl-TRel-weakly-respects-barbs[where TRel=TRel]
    weak-respection-of-barbs-and-closures(5)[where Rel=TRel and CWB=TBW]
    by blast
qed

lemma (in encoding-wrt-barbs) indRelRTPO-impl-TRel-is-strong-barbed-bisimulation:
  fixes TRel :: ('procT × 'procT) set
  assumes bisim: strong-barbed-bisimulation (indRelRTPO TRel) (STCalWB SWB TBW)
  shows strong-barbed-bisimulation (TRel+) TBW
proof
  from bisim show strong-reduction-bisimulation (TRel+) (Calculus TBW)
5.3 Relations Induced by the Encoding and Relations on Source Terms and Target Terms

Some encodability like e.g. full abstraction are defined w.r.t. a relation on source terms and a relation on target terms. To analyse such criteria we include these two relations in the considered relation on the disjoint union of source and target terms.

**Inductive-set (in encoding) indRelRST**

\[ \text{indRelRST} : (\langle\text{proc}\ S \times \text{proc}\ S\rangle) \Rightarrow (\langle\text{proc}\ T \times \text{proc}\ T\rangle) \Rightarrow (\langle\text{proc}\ \mathit{Proc}\ \text{Proc}\ \times \langle\text{proc}\ \mathit{Proc}\ \text{Proc}\rangle\text{Proc}\rangle)\text{ set} \]

**for SRel : (\langle\text{proc}\ S \times \text{proc}\ S\rangle) set**

**and TRel : (\langle\text{proc}\ T \times \text{proc}\ T\rangle) set**

**where**

\[ \text{encR} : (\text{SourceTerm } S, \text{TargetTerm } (\llbracket S \rrbracket)) \in \text{indRelRST } S\text{Rel } T\text{Rel} | \]

**source: (S1, S2) ∈ SRel \Rightarrow (\text{SourceTerm } S1, \text{SourceTerm } S2) \in \text{indRelRST } S\text{Rel } T\text{Rel} | \]

**target: (T1, T2) ∈ TRel \Rightarrow (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{indRelRST } S\text{Rel } T\text{Rel} | \]

**abbreviation (in encoding) indRelRSTinfix**

\[ \text{indRelRSTinfix} : (\langle\text{proc}\ S, \text{proc}\ T\rangle) \text{Proc} \Rightarrow (\langle\text{proc}\ S \times \text{proc}\ S\rangle) \text{set} \Rightarrow (\langle\text{proc}\ T \times \text{proc}\ T\rangle) \text{set} \Rightarrow (\langle\text{proc}\ S, \text{proc}\ T\rangle) \text{Proc} \Rightarrow \text{bool} (-\ R\ ] R<-.-> -\ 75, 75, 75, 75\ ] 80) \]

**where**

\[ P\ R\ ] R<SRRel,TRel> Q \equiv (P, Q) \in \text{indRelRST } S\text{Rel } T\text{Rel} \]

```
inductive-set (in encoding) indRelRSTPO
:: ('procS × 'procS) set ⇒ ('procT × 'procT) set
for SRel :: ('procS × 'procS) set
and TRel :: ('procT × 'procT) set
where
encR: (SourceTerm S, TargetTerm ([S])) ∈ indRelRSTPO SRel TRel |
source: (S1, S2) ∈ SRel ⇒ (SourceTerm S1, SourceTerm S2) ∈ indRelRSTPO SRel TRel |
target: (T1, T2) ∈ TRel ⇒ (TargetTerm T1, TargetTerm T2) ∈ indRelRSTPO SRel TRel |
trans: [(P, Q) ∈ indRelRSTPO SRel TRel; (Q, R) ∈ indRelRSTPO SRel TRel] ⇒ (P, R) ∈ indRelRSTPO SRel TRel

abbreviation (in encoding) indRelRSTPOinfix ::
('procS, 'procT) Proc ⇒ ('procS × 'procS) set ⇒ ('procT × 'procT) set
⇒ ('procS, 'procT) Proc ⇒ bool (- ≤[-]R<,-,-> - [75, 75, 75, 75] 80)
where
P ≤[-]R<SRel,TRel> Q ≡ (P, Q) ∈ indRelRSTPO SRel TRel

lemma (in encoding) indRelRSTPO-refl:
fixes SRel :: ('procS × 'procS) set
and TRel :: ('procT × 'procT) set
assumes reflS: refl SRel
and reflT: refl TRel
shows refl (indRelRSTPO SRel TRel)
unfolding refl-on-def
proof auto
  fix P
  show P ≤[-]R<SRel,TRel> P
  proof (cases P)
    case (SourceTerm SP)
    assume SP ∈ S P
    with reflS show P ≤[-]R<SRel,TRel> P
    unfolding refl-on-def
    by (simp add: indRelRSTPO.source)
  next
    case (TargetTerm TP)
    assume TP ∈ T P
    with reflT show P ≤[-]R<SRel,TRel> P
    unfolding refl-on-def
    by (simp add: indRelRSTPO.target)
  qed
qed

lemma (in encoding) indRelRSTPO-trans:
fixes SRel :: ('procS × 'procS) set
and TRel :: ('procT × 'procT) set
shows trans (indRelRSTPO SRel TRel)
unfolding trans-def
proof clarify
  fix P Q R
  assume P ≤[-]R<SRel,TRel> Q and Q ≤[-]R<SRel,TRel> R
  thus P ≤[-]R<SRel,TRel> R
  by (rule indRelRSTPO.trans)
qed

lemma (in encoding) refl-trans-closure-of-indRelRST:
fixes SRel :: ('procS × 'procS) set
and TRel :: ('procT × 'procT) set
assumes reflS: refl SRel
and reflT: refl TRel
shows \(\text{indRelRSTPO} \ S \ R \ T \ R = (\text{indRelRST} \ S \ R \ T \ R)^*\)

proof auto
fix \(P\)
assume \(P \leq [\cdot] R < S \ R, T \ R\) \(\rightarrow\) \(Q\)
thus \((P, Q) \in (\text{indRelRST} \ S \ R \ T \ R)^*\)
proof induct
\(\text{case} (\text{encL} \ S)\)
\(\text{show} \ (\text{SourceTerm} \ S, \text{TargetTerm} \ ([S])) \in (\text{indRelRST} \ S \ R \ T \ R)^*\)
\(\text{using} \ \text{indRelRST.}\text{encR[of S S \ R \ T \ R]}\)
\(\text{by simp}\)
next
\(\text{case} (\text{source} \ S1 \ S2)\)
\(\text{assume} \ (S1, S2) \in S \ R \ T \ R\)
\(\text{thus} \ (\text{SourceTerm} \ S1, \text{SourceTerm} \ S2) \in (\text{indRelRST} \ S \ R \ T \ R)^*\)
\(\text{using} \ \text{indRelRST.}\text{source[of S1 S2 S \ R \ T \ R]}\)
\(\text{by simp}\)
next
\(\text{case} (\text{target} \ T1 \ T2)\)
\(\text{assume} \ (T1, T2) \in T \ R \ T \ R\)
\(\text{thus} \ (\text{TargetTerm} \ T1, \text{TargetTerm} \ T2) \in (\text{indRelRST} \ S \ R \ T \ R)^*\)
\(\text{using} \ \text{indRelRST.}\text{target[of T1 T2 T \ R \ S \ R]}\)
\(\text{by simp}\)
next
\(\text{case} (\text{trans} \ P \ Q \ R)\)
\(\text{assume} \ (P, Q) \in (\text{indRelRST} \ S \ R \ T \ R)^* \ \text{and} \ (Q, R) \in (\text{indRelRST} \ S \ R \ T \ R)^*\)
\(\text{thus} \ (P, R) \in (\text{indRelRST} \ S \ R \ T \ R)^*\)
\(\text{by simp}\)
qed
next
fix \(P\)
assume \((P, Q) \in (\text{indRelRST} \ S \ R \ T \ R)^*\)
thus \(P \leq [\cdot] R < S \ R, T \ R\) \(\rightarrow\) \(Q\)
proof induct
\(\text{from reflS reflux show} \ P \leq [\cdot] R < S \ R, T \ R\) \(\rightarrow\) \(P\)
\(\text{using} \ \text{indRelRSTPO-refl[of S R T R]}\)
\(\text{unfolding} \ \text{refl-on-def}\)
\(\text{by simp}\)
next
\(\text{case} (\text{step} \ Q \ R)\)
\(\text{assume} \ P \leq [\cdot] R < S \ R, T \ R\) \(\rightarrow\) \(Q\)
moreover assume \(Q \leq [\cdot] R < S \ R, T \ R\) \(\rightarrow\) \(R\)
\(\text{hence} \ Q \leq [\cdot] R < S \ R, T \ R\) \(\rightarrow\) \(R\)
\(\text{by (induct, simp-all add: indRelRSTPO.intros)}\)
\(\text{ultimately show} \ P \leq [\cdot] R < S \ R, T \ R\) \(\rightarrow\) \(R\)
\(\text{by (rule indRelRSTPO.trans)}\)
qed
qed

inductive-set (in encoding) \text{indRelLST}\n:: \((\text{procS} \times \text{procS}) \ \text{set} \Rightarrow (\text{procT} \times \text{procT}) \ \text{set})
\Rightarrow ((\text{procS} \times \text{procT}) \ \text{Proc}) \times ((\text{procS} \times \text{procT}) \ \text{Proc}) \ \text{set})
for \text{SRel} :: ((\text{procS} \times \text{procS}) \ \text{set})
and \text{TRel} :: ((\text{procT} \times \text{procT}) \ \text{set})
where
\text{encL:} \ (\text{TargetTerm} \ ([S]), \text{SourceTerm} \ S) \in \text{indRelLST} \ \text{Srel} \ \text{Trel} \ |
\text{source:} \ (S1, S2) \in \text{Srel} \Rightarrow \ (\text{SourceTerm} \ S1, \text{SourceTerm} \ S2) \in \text{indRelLST} \ \text{Srel} \ \text{Trel} \ |
\text{target:} \ (T1, T2) \in \text{Trel} \Rightarrow \ (\text{TargetTerm} \ T1, \text{TargetTerm} \ T2) \in \text{indRelLST} \ \text{Srel} \ \text{Trel}

abbreviation (in encoding) \text{indRelLST} infix
:: \((\text{procS} \times \text{procT}) \ \text{Proc} \Rightarrow (\text{procS} \times \text{procS}) \ \text{set} \Rightarrow (\text{procT} \times \text{procT}) \ \text{set})
\Rightarrow ((\text{procS} \times \text{procT}) \ \text{Proc} \Rightarrow \text{bool} \ (- \ R \ [\cdot] L \leq \cdot \rightarrow \cdot \ [75, 75, 75, 75] 80))
where
\( P \mathcal{R} \mathcal{I} \mathcal{I} L < S R e l, T R e l > Q \equiv (P, Q) \in \text{indRelLSTPO} S R e l T R e l \)

**inductive-set (in encoding) indRelLSTPO**

\[
\begin{align*}
&\text{:: ('procS} \times \text{'procS} set \Rightarrow ('procT} \times \text{'procT} set \\
&\Rightarrow ((\text{'procS} \times \text{'procT}) \text{Proc}) \times ((\text{'procS} \times \text{'procT}) \text{Proc}) set \\
&\text{for } S R e l \equiv ('procS} \times \text{'procS} set \\
&\text{and } T R e l \equiv ('procT} \times \text{'procT} set \\
&\text{where}
\end{align*}
\]

\[
\begin{align*}
&\text{encL: (TargetTerm ([S]), SourceTerm S) } \in \text{indRelLSTPO SRel TRel |} \\
&\text{source: (S1, S2) } \in S R e l \Rightarrow (SourceTerm S1, SourceTerm S2) \in \text{indRelLSTPO SRel TRel |} \\
&\text{target: (T1, T2) } \in T R e l \Rightarrow (TargetTerm T1, TargetTerm T2) \in \text{indRelLSTPO SRel TRel |} \\
&\text{trans: } [(P, Q) \in \text{indRelLSTPO SRel TRel; (Q, R) } \in \text{indRelLSTPO SRel TRel}] \\
&\Rightarrow (P, R) \in \text{indRelLSTPO SRel TRel}
\end{align*}
\]

**abbreviation (in encoding) indRelLSTPOinfix**

\[
\begin{align*}
&\text{:: ('procS} \times \text{'procT}) \text{Proc} \Rightarrow ('procS} \times \text{'procS} set \Rightarrow ('procT} \times \text{'procT} set \\
&\Rightarrow ('procS} \times \text{'procT}) \text{Proc} \Rightarrow \text{bool} (- \leq \nmid L < -, > - [75, 75, 75, 75] 80) \\
&\text{where}
\end{align*}
\]

\[
\begin{align*}
&P \leq \nmid L < S R e l, T R e l > Q \equiv (P, Q) \in \text{indRelLSTPO SRel TRel}
\end{align*}
\]

**lemma (in encoding) indRelLSTPO-refl**

\[
\begin{align*}
&\text{fixes } S R e l \equiv ('procS} \times \text{'procS} set \\
&\text{and } T R e l \equiv ('procT} \times \text{'procT} set \\
&\text{assumes reflS: refl SRel} \\
&\text{and reflT: refl TRel} \\
&\text{shows refl (indRelLSTPO SRel TRel)} \\
&\text{unfolding refl-on-def}
\end{align*}
\]

**proof auto**

\[
\begin{align*}
&\text{fix } P \\
&\text{show } P \leq \nmid L < S R e l, T R e l > P \\
&\text{proof (cases } P) \\
&\text{case (SourceTerm SP) } \\
&\text{assume SP } \in S P \\
&\text{with reflS show } P \leq \nmid L < S R e l, T R e l > P \\
&\text{unfolding refl-on-def} \\
&\text{by (simp add: indRelLSTPO.source)} \\
&\text{next} \\
&\text{case (TargetTerm TP) } \\
&\text{assume TP } \in T P \\
&\text{with reflT show } P \leq \nmid L < S R e l, T R e l > P \\
&\text{unfolding refl-on-def} \\
&\text{by (simp add: indRelLSTPO.target)} \\
&\text{qed}
\end{align*}
\]

**lemma (in encoding) indRelLSTPO-trans**

\[
\begin{align*}
&\text{fixes } S R e l \equiv ('procS} \times \text{'procS} set \\
&\text{and } T R e l \equiv ('procT} \times \text{'procT} set \\
&\text{shows trans (indRelLSTPO SRel TRel)} \\
&\text{unfolding trans-def}
\end{align*}
\]

**proof clarify**

\[
\begin{align*}
&\text{fix } P Q R \\
&\text{assume } P \leq \nmid L < S R e l, T R e l > Q \text{ and } Q \leq \nmid L < S R e l, T R e l > R \\
&\text{thus } P \leq \nmid L < S R e l, T R e l > R \\
&\text{by (rule indRelLSTPO.trans)} \\
&\text{qed}
\end{align*}
\]

**lemma (in encoding) refl-trans-closure-of-indRelLST**

\[
\begin{align*}
&\text{fixes } S R e l \equiv ('procS} \times \text{'procS} set \\
&\text{and } T R e l \equiv ('procT} \times \text{'procT} set
\end{align*}
\]

123
assumes reflS: refl SRel and reflT: refl TRel
shows indRelLSTPO SRel TRel = (indRelLST SRel TRel)^* 
proof auto 
fix P Q
assume P ≤ις L< SRel, TRel> Q
thus (P, Q) ∈ (indRelLST SRel TRel)^* 
proof induct
  case (encL S)
  show (TargetTerm ([S]), SourceTerm S) ∈ (indRelLST SRel TRel)^* using indRelLST.encL[of S SRel TRel]
  by simp
next
case (source S1 S2)
assume (S1, S2) ∈ SRel
thus (SourceTerm S1, SourceTerm S2) ∈ (indRelLST SRel TRel)^* using indRelLST.source[of S1 S2 SRel TRel]
by simp
next
case (target T1 T2)
assume (T1, T2) ∈ TRel
thus (TargetTerm T1, TargetTerm T2) ∈ (indRelLST SRel TRel)^* using indRelLST.target[of T1 T2 TRel SRel]
by simp
next
case (trans P Q R)
assume (P, Q) ∈ (indRelLST SRel TRel)^* and (Q, R) ∈ (indRelLST SRel TRel)^*
thus (P, R) ∈ (indRelLST SRel TRel)^* by simp
qed
next
fix P Q
assume (P, Q) ∈ (indRelLST SRel TRel)^* 
thus P ≤ις L< SRel, TRel> Q
proof induct
  from reflS reflT show P ≤ις L< SRel, TRel> P
  unfolding refl-on-def
  by simp
next
case (step Q R)
assume P ≤ις L< SRel, TRel> Q
moreover assume Q ≤ις L< SRel, TRel> R
hence Q ≤ις L< SRel, TRel> R
  by (induct, simp-all add: indRelLSTPO.intros)
ultimately show P ≤ις L< SRel, TRel> R
  by (rule indRelLSTPO.trans)
qed

inductive-set (in encoding) indRelST :: ('procS × 'procS) set ⇒ ('procT × 'procT) set
  for SRel :: ('procS × 'procS) set
  and TRel :: ('procT × 'procT) set
where
  encR: (SourceTerm S, TargetTerm ([S])) ∈ indRelST SRel TRel |
  encL: (TargetTerm ([S]), SourceTerm S) ∈ indRelST SRel TRel |
  source: (S1, S2) ∈ SRel ⇒ (SourceTerm S1, SourceTerm S2) ∈ indRelST SRel TRel |
  target: (T1, T2) ∈ TRel ⇒ (TargetTerm T1, TargetTerm T2) ∈ indRelST SRel TRel
abbreviation \textbf{(in \ encoding)} \texttt{indRelSTinfix} \\
\texttt{:: \ ('procS \times \ 'procS) Proc ⇒ ('procT \times \ 'procT) set} \\
\texttt{⇒ ('procS \times \ 'procT) Proc ⇒ bool (- \ \mathcal{R}, [-75, 75, 75, 75] 80)} \\
\texttt{where} \\
P \mathcal{R} \equiv \langle SRel, TRel \rangle \ Q \equiv (P, Q) \in \text{indRelST SRel TRel}

\textbf{lemma (in \ encoding)} \texttt{indRelST-symm:} \\
\texttt{\textbf{fixes} SRel :: ('procS \times \ 'procS) set} \\
\texttt{\textbf{and} TRel :: ('procT \times \ 'procT) set} \\
\texttt{\textbf{assumes} symmS: \ sym \ SRel} \\
\texttt{\textbf{and} symmT: \ sym \ TRel} \\
\texttt{\textbf{shows} sym (indRelST SRel TRel)} \\
\texttt{\textbf{unfolding} sym-def} \\
\texttt{\textbf{proof} clarify} \\
\texttt{\textbf{fix} P Q} \\
\texttt{\textbf{assume} (P, Q) \in \text{indRelST SRel TRel}} \\
\texttt{\textbf{thus} (Q, P) \in \text{indRelST SRel TRel}} \\
\texttt{\textbf{proof induct}} \\
\texttt{\textbf{case} (encR S)} \\
\texttt{show (TargetTerm \ ([S]), SourceTerm S) \in \text{indRelST SRel TRel}} \\
\texttt{by (rule indRelST.encL)} \\
\texttt{next} \\
\texttt{\textbf{case} (encL S)} \\
\texttt{show (SourceTerm S, TargetTerm \ ([S])) \in \text{indRelST SRel TRel}} \\
\texttt{by (rule indRelST.encR)} \\
\texttt{next} \\
\texttt{\textbf{case} (source S1 S2)} \\
\texttt{\textbf{assume} (S1, S2) \in \text{SRel}} \\
\texttt{\textbf{with} symmS show (SourceTerm S2, SourceTerm S1) \in \text{indRelST SRel TRel}} \\
\texttt{\textbf{unfolding} sym-def} \\
\texttt{by (simp add: \text{indRelST.source})} \\
\texttt{next} \\
\texttt{\textbf{case} (target T1 T2)} \\
\texttt{\textbf{assume} (T1, T2) \in \text{TRel}} \\
\texttt{\textbf{with} symmT show (TargetTerm T2, TargetTerm T1) \in \text{indRelST SRel TRel}} \\
\texttt{\textbf{unfolding} sym-def} \\
\texttt{by (simp add: \text{indRelST.target})} \\
\texttt{qed} \\
\texttt{\textbf{qed}}

\textbf{inductive-set (in \ encoding)} \texttt{indRelSTEQ} \\
\texttt{:: ('procS \times \ 'procS) set ⇒ ('procT \times \ 'procT) set} \\
\texttt{⇒ (((\ 'procS, \ 'procT) Proc \times ((\ 'procS, \ 'procT) Proc)) set} \\
\texttt{\textbf{for} SRel :: ('procS \times \ 'procS) set} \\
\texttt{\textbf{and} TRel :: ('procT \times \ 'procT) set} \\
\texttt{\textbf{where}} \\
\texttt{encR: \ (SourceTerm S, TargetTerm \ ([S])) \in \text{indRelSTEQ SRel TRel} \mid} \\
\texttt{encL: \ (TargetTerm \ ([S]), SourceTerm S) \in \text{indRelSTEQ SRel TRel} \mid} \\
\texttt{source: \ (S1, S2) \in \text{SRel} \Rightarrow (SourceTerm S1, SourceTerm S2) \in \text{indRelSTEQ SRel TRel} \mid} \\
\texttt{target: \ (T1, T2) \in \text{TRel} \Rightarrow (TargetTerm T1, TargetTerm T2) \in \text{indRelSTEQ SRel TRel} \mid} \\
\texttt{trans: \ \langle P, Q \rangle \in \text{indRelSTEQ SRel TRel;} \ (Q, R) \in \text{indRelSTEQ SRel TRel} \mid} \\
\texttt{\Rightarrow \ (P, R) \in \text{indRelSTEQ SRel TRel}} \\
\texttt{abbreviation (in \ encoding)} \texttt{indRelSTEQinfix} \\
\texttt{:: ('procS, 'procT) Proc ⇒ ('procS \times \ 'procS) set} \\
\texttt{⇒ ('procS, 'procT) Proc ⇒ bool (- \ \sim [-75, 75, 75, 75] 80)} \\
\texttt{where} \\
P \sim \equiv \langle SRel, TRel \rangle \ Q \equiv (P, Q) \in \text{indRelSTEQ SRel TRel}

\textbf{lemma (in \ encoding)} \texttt{indRelSTEQ-refl:} \\
\texttt{\textbf{fixes} SRel :: ('procS \times \ 'procS) set}
and \( T\text{Rel} \) :: \( (\text{proc} T \times \text{proc} T) \) set
assumes refl\( T \) : refl \( T\text{Rel} \)
shows refl (ind\( T\text{RelSTEQ} S\text{Rel} T\text{Rel} \)
\[ \text{unfolding refl-on-def} \]
\[ \text{proof auto} \]
fix \( P \)
show \( P \sim [\[\cdot\]] <S\text{Rel}, T\text{Rel}> P \)
\[ \text{proof (cases} P \) \]
case (SourceTerm \( SP \))
assume \( SP \in S P \)
moreover have SourceTerm \( SP \sim [\[\cdot\]] <S\text{Rel}, T\text{Rel}> \) TargetTerm \( ([SP]) \)
by (rule ind\( T\text{RelSTEQ}.\text{encR} \))
moreover have TargetTerm \( ([SP]) \sim [\[\cdot\]] <S\text{Rel}, T\text{Rel}> \) SourceTerm \( SP \)
by (rule ind\( T\text{RelSTEQ}.\text{encL} \))
ultimately show \( P \sim [\[\cdot\]] <S\text{Rel}, T\text{Rel}> P \)
by (simp add: \( \text{ind\( T\text{RelSTEQ}.\text{trans} \), where} \) \( P = \) SourceTerm \( SP \) \text{ and } Q = \) TargetTerm \( ([SP]) \))
next
case (TargetTerm \( TP \))
assume \( TP \in T P \)
with refl\( T \) show \( P \sim [\[\cdot\]] <S\text{Rel}, T\text{Rel}> P \)
\[ \text{unfolding refl-on-def} \]
by (simp add: \( \text{ind\( T\text{RelSTEQ}.\text{target} \)})
qed
qed

\[ \text{lemma (in encoding)} \text{\( \text{ind\( T\text{RelSTEQ-symm}\):} \]
fixes \( S\text{Rel} \) :: \( (\text{proc} S \times \text{proc} S) \) set
\[ \text{and } T\text{Rel} : (\text{proc} T \times \text{proc} T) \) set
assumes symm\( S \) : sym \( S\text{Rel} \)
\[ \text{and symm\( T \) : sym \( T\text{Rel} \)} \]
shows sym (ind\( T\text{RelSTEQ} S\text{Rel} T\text{Rel} \)
\[ \text{unfolding sym-def} \]
\[ \text{proof clarify} \]
fix \( P \) \( Q \)
assume \( P \sim [\[\cdot\]] <S\text{Rel}, T\text{Rel}> Q \)
thus \( Q \sim [\[\cdot\]] <S\text{Rel}, T\text{Rel}> P \)
\[ \text{proof induct} \]
case (encR \( S \))
show TargetTerm \( ([S]) \sim [\[\cdot\]] <S\text{Rel}, T\text{Rel}> \) SourceTerm \( S \)
by (rule ind\( T\text{RelSTEQ}.\text{encL} \))
next
case (encL \( S \))
show SourceTerm \( S \sim [\[\cdot\]] <S\text{Rel}, T\text{Rel}> \) TargetTerm \( ([S]) \)
by (rule ind\( T\text{RelSTEQ}.\text{encR} \))
next
case (source \( S1 \) \( S2 \))
assume \( (S1, S2) \in S\text{Rel} \)
with symm\( S \) show SourceTerm \( S2 \sim [\[\cdot\]] <S\text{Rel}, T\text{Rel}> \) SourceTerm \( S1 \)
\[ \text{unfolding sym-def} \]
by (simp add: \( \text{ind\( T\text{RelSTEQ}.\text{source} \)})
next
case (target \( T1 \) \( T2 \))
assume \( (T1, T2) \in T\text{Rel} \)
with symm\( T \) show TargetTerm \( T2 \sim [\[\cdot\]] <S\text{Rel}, T\text{Rel}> \) TargetTerm \( T1 \)
\[ \text{unfolding sym-def} \]
by (simp add: \( \text{ind\( T\text{RelSTEQ}.\text{target} \)})
next
case (trans \( P \) \( Q \) \( R \))
assume \( R \sim [\[\cdot\]] <S\text{Rel}, T\text{Rel}> Q \) \text{ and } \( Q \sim [\[\cdot\]] <S\text{Rel}, T\text{Rel}> P \)
thus \( R \sim [\[\cdot\]] <S\text{Rel}, T\text{Rel}> P \)
by (rule ind\( T\text{RelSTEQ}.\text{trans} \))
\[ 126 \]
lemma (in encoding) indRelSTEQ-trans:
fixes SRel :: ('procS × 'procS) set
and TRel :: ('procT × 'procT) set
shows trans (indRelSTEQ SRel TRel)
unfolding trans-def
proof clarify
fix P Q R
assume \(P \sim SRel, TRel > Q\) and \(Q \sim SRel, TRel > R\)
thus \(P \sim SRel, TRel > R\)
by (rule indRelSTEQ.trans)
qed

lemma (in encoding) refl-trans-closure-of-indRelST:
fixes SRel :: ('procS × 'procS) set
and TRel :: ('procT × 'procT) set
assumes reflT: refl TRel
shows indRelSTEQ SRel TRel = (indRelST SRel TRel)*
proof auto
fix P Q
assume \(P \sim SRel, TRel > Q\)
thus \((P, Q) \in (indRelST SRel TRel)*\)
proof induct
  case (encR S)
  show \((\text{SourceTerm } S, \text{TargetTerm } [S]) \in (\text{indRelST SRel TRel})*\)
  using indRelST.encR[of S SRel TRel]
  by simp
next
  case (encL S)
  show \((\text{TargetTerm } [S], \text{SourceTerm } S) \in (\text{indRelST SRel TRel})*\)
  using indRelST.encL[of S SRel TRel]
  by simp
next
  case (source S1 S2)
  assume \((S1, S2) \in SRel\)
  thus \((\text{SourceTerm } S1, \text{SourceTerm } S2) \in (\text{indRelST SRel TRel})*\)
  using indRelST.source[of S1 S2 SRel TRel]
  by simp
next
  case (target T1 T2)
  assume \((T1, T2) \in TRel\)
  thus \((\text{TargetTerm } T1, \text{TargetTerm } T2) \in (\text{indRelST SRel TRel})*\)
  using indRelST.target[of T1 T2 TRel SRel]
  by simp
next
  case (trans P Q R)
  assume \((P, Q) \in (\text{indRelST SRel TRel})*\) and \((Q, R) \in (\text{indRelST SRel TRel})*\)
  thus \((P, R) \in (\text{indRelST SRel TRel})*\)
  by simp
qed
next
fix P Q
assume \((P, Q) \in (\text{indRelST SRel TRel})*\)
thus \(P \sim SRel, TRel > Q\)
proof induct
from reflT show \(P \sim SRel, TRel > P\)
  using indRelSTEQ-refl[of TRel SRel]
  unfolding refl-on-def
  by simp
next

case (step Q R)
assume \( P \sim[S, T]Q \)
moreover assume \( Q \sim[S, T]R \)

hence \( Q \sim[S, T]R \)
  by (induct, simp-all add: indRelSTEQ.intros)
ultimately show \( P \sim[S, T]R \)
  by (rule indRelSTEQ.trans)

qed

lemma (in encoding) refl-symm-trans-closure-of-indRelST:
  fixes SRel :: ('procS × 'procS) set
  and TRel :: ('procT × 'procT) set
  assumes reflT: refl TRel
  and symmS: symm SRel
  and symmT: symm TRel
  shows indRelSTEQ SRel TRel = (symcl ((indRelST SRel TRel)\(\equiv\))\(\approx\))\(\approx\)
proof	have (symcl ((indRelST SRel TRel)\(\equiv\))\(\approx\))\(\approx\) = (symcl (indRelST SRel TRel)\(\equiv\))\(\approx\)
  by (rule refl-symm-trans-closure-is-symm-refl-trans-closure[where Rel=indRelST SRel TRel])
moreover have symm\(\approx\) symm\(\approx\) TRel have symcl (indRelST SRel TRel) = indRelST SRel TRel
  using indRelST-symm[where SRel=SRel and TRel=TRel]
  symm-closure-of-symm-rel[where Rel=indRelST SRel TRel]
  by blast
ultimately show indRelSTEQ SRel TRel = (symcl ((indRelST SRel TRel)\(\equiv\))\(\approx\))\(\approx\)
  using reflT refl-trans-closure-of-indRelST[where SRel=SRel and TRel=TRel]
  by simp

qed

lemma (in encoding) symm-closure-of-indRelRST:
  fixes SRel :: ('procS × 'procS) set
  and TRel :: ('procT × 'procT) set
  assumes reflT: refl TRel
  and symmS: symm SRel
  and symmT: symm TRel
  shows indRelRST SRel TRel = symcl (indRelRST SRel TRel)
  and indRelSTEQ SRel TRel = (symcl ((indRelRST SRel TRel)\(\equiv\))\(\approx\))\(\approx\)
proof
  show indRelRST SRel TRel = symcl (indRelRST SRel TRel)
proof auto
fix P Q
assume \( P \sim[S, T]Q \)
thus \((P, Q) \in symcl (indRelRST SRel TRel)\)
  by (induct, simp-all add: symcl-def indRelRST.intros)

next
fix P Q
assume \((P, Q) \in symcl (indRelRST SRel TRel)\)
thus \( P \sim[S, T]Q \)
proof (auto simp add: symcl-def indRelRST.simps)
fix S
show SourceTerm S \(\sim[S, T]\) TargetTerm (\(\{S\}\))
  by (rule indRelST.encR)
next
fix S1 S2
assume \((S1, S2) \in SRel\)
thus SourceTerm S1 \(\sim[S, T]\) SourceTerm S2
  by (rule indRelST.source)
next
fix T1 T2
assume \((T1, T2) \in TRel\)
thus \( \text{TargetTerm} \ T_1 \ R \ [\cdot] \ <SRel, TRel> \ \text{TargetTerm} \ T_2 \)
by (rule \( \text{indRelST} \).target)

next
fix \( S \)
show \( \text{TargetTerm} \ ([S]) \ R \ [\cdot] \ <SRel, TRel> \ \text{SourceTerm} \ S \)
by (rule \( \text{indRelST} \).encL)

next
fix \( S_1 \ S_2 \)
assume \( (S_1, S_2) \in SRel \)
with \( \textit{symmS} \) show \( \text{SourceTerm} \ S_2 \ R \ [\cdot] \ <SRel, TRel> \ \text{SourceTerm} \ S_1 \)

unfolding \( \text{sym-def} \)
by (simp add: \( \text{indRelST} \).source)

next
fix \( T_1 \ T_2 \)
assume \( (T_1, T_2) \in TRel \)
with \( \textit{symmT} \) show \((\text{TargetTerm} \ T_2, \text{TargetTerm} \ T_1) \in \text{indRelST} \ SRel \ TRel \)
unfolding \( \text{sym-def} \)
by (simp add: \( \text{indRelST} \).target)

qed

definition \( \text{indRelSTEQ} \) \( SRel \ TRel \equiv \) \( \text{symcl}(\text{indRelRST} \ SRel \ TRel) \)
using \( \text{refl-symm-trans-closure-is-symm-refl-trans-closure} \) \[ \text{where} \ Rel=\text{indRelRST} \ SRel \ TRel \]

proof -
show \( \text{indRelST} \ SRel \ TRel = \text{symcl}(\text{indRelLST} \ SRel \ TRel) \)
and \( \text{indRelSTEQ} \ SRel \ TRel = (\text{symcl}(\text{indRelLST} \ SRel \ TRel)^+) \)

proof auto
fix \( P \ Q \)
assume \( P \ R \ [\cdot] \ <SRel, TRel> \ Q \)
thus \( (P, Q) \in \text{symcl}(\text{indRelLST} \ SRel \ TRel) \)
by (induct, simp-all add: \( \text{symcl-def} \ \text{indRelLST-intros} \))

next
fix \( P \ Q \)
assume \( (P, Q) \in \text{symcl}(\text{indRelLST} \ SRel \ TRel) \)
thus \( P \ R \ [\cdot] \ <SRel, TRel> \ Q \)
proof (auto simp add: \( \text{symcl-def} \ \text{indRelLST-simps} \))
fix \( S \)
show \( \text{SourceTerm} \ S \ R \ [\cdot] \ <SRel, TRel> \ \text{TargetTerm} \ ([S]) \)
by (rule \( \text{indRelST} \).encL)

next
fix \( S_1 \ S_2 \)
assume \( (S_1, S_2) \in SRel \)
thus \( \text{SourceTerm} \ S_1 \ R \ [\cdot] \ <SRel, TRel> \ \text{SourceTerm} \ S_2 \)
by (rule \( \text{indRelST} \).source)

next
fix \( T_1 \ T_2 \)
assume \( (T_1, T_2) \in TRel \)
thus \( \text{TargetTerm} \ T_1 \ R \ [\cdot] \ <SRel, TRel> \ \text{TargetTerm} \ T_2 \)
by (rule \( \text{indRelST} \).target)

next
fix \( S \)
\begin{verbatim}
show TargetTerm ([S]) R[S][<SRel,TRel>] SourceTerm S
  by (rule indRelST_encL)
next
fix S1 S2
assume (S1, S2) ∈ SRel
with symmS show SourceTerm S2 R[S]<SRel,TRel> SourceTerm S1
  unfolding sym-def
  by (simp add: indRelST_source)
next
fix T1 T2
assume (T1, T2) ∈ TRel
with symmT show TargetTerm T2 R[T]<SRel,TRel> TargetTerm T1
  unfolding sym-def
  by (simp add: indRelST_target)
qed
qed
with reflT show indRelSTEQ SRel TRel = (symcl((indRelLST SRel TRel)\^{}))
  using refl-symm-trans-closure-is-symm-refl-trans-closure
    where Rel = indRelLST SRel TRel
by simp qed

lemma (in encoding) symm-trans-closure-of-indRelRSTPO:
  fixes SRel :: ('procS × 'procS) set
  and TRel :: ('procT × 'procT) set
  assumes symmS: sym SRel
  and symmT: sym TRel
  shows indRelSTEQ SRel TRel = (symcl(indRelRSTPO SRel TRel))
proof auto
fix P Q
assume P ~<SRel,TRel> Q
thus (P, Q) ∈ (symcl(indRelRSTPO SRel TRel))
proof induct
  case (encR S)
  show (SourceTerm S, TargetTerm ([S])) ∈ (symcl(indRelRSTPO SRel TRel))
    using indRelRSTPO_encR[of S SRel TRel]
    unfolding symcl-def
    by auto
next
  case (encL S)
  show (TargetTerm ([S]), SourceTerm S) ∈ (symcl(indRelRSTPO SRel TRel))
    using indRelRSTPO_encL[of S SRel TRel]
    unfolding symcl-def
    by auto
next
  case (source S1 S2)
  assume (S1, S2) ∈ SRel
  thus (SourceTerm S1, SourceTerm S2) ∈ (symcl(indRelRSTPO SRel TRel))
    using indRelRSTPO_source[of S1 S2 SRel TRel]
    unfolding symcl-def
    by auto
next
  case (target T1 T2)
  assume (T1, T2) ∈ TRel
  thus (TargetTerm T1, TargetTerm T2) ∈ (symcl(indRelRSTPO SRel TRel))
    using indRelRSTPO_target[of T1 T2 TRel SRel]
    unfolding symcl-def
    by auto
next
  case (trans P Q R)
  assume (P, Q) ∈ (symcl(indRelRSTPO SRel TRel))
\end{verbatim}
and $(Q, R) \in (\text{symcl}(\text{indRelRSTPO} \ SRel \ TRel))^+$
thus $(P, R) \in (\text{symcl}(\text{indRelRSTPO} \ SRel \ TRel))^+$
by simp
qed

next
fix $P \ Q$
assume $(P, Q) \in (\text{symcl}(\text{indRelRSTPO} \ SRel \ TRel))^+$
thus $P \sim [\cdot]_{SRel, \ TRel} \ Q$
proof induct
fix $Q$
assume $(P, Q) \in (\text{symcl}(\text{indRelRSTPO} \ SRel \ TRel))$
thus $P \sim [\cdot]_{SRel, \ TRel} \ Q$
proof
(cases $P \preceq [\cdot]_{SRel, \ TRel} \ Q$ simp-all add: symcl-def)
assume $P \preceq [\cdot]_{SRel, \ TRel} \ Q$
thus $P \sim [\cdot]_{SRel, \ TRel} \ Q$
proof induct

next
assume $Q \preceq [\cdot]_{SRel, \ TRel} \ P$
thus $P \sim [\cdot]_{SRel, \ TRel} \ Q$
proof
(cases $P \preceq [\cdot]_{SRel, \ TRel} \ Q$ simp-all add: symcl-def)
assume $P \preceq [\cdot]_{SRel, \ TRel} \ Q$
thus $P \sim [\cdot]_{SRel, \ TRel} \ Q$
proof induct

qed
qed

next
next
  case (step Q R)
  assume \( P \sim [\cdot] < SRel, TRel > Q \)
  moreover assume \((Q, R) \in \text{symcl} (\text{indRelRSTPO} SRel TRel)\)
  hence \( Q \sim [\cdot] < SRel, TRel > R \)
  proof (auto simp add: symcl-def)
    assume \( Q \leq [\cdot] R < SRel, TRel > R \)
    thus \( Q \sim [\cdot] < SRel, TRel > R \)
  proof (induct, simp add: indRelSTEQ.encR, simp add: indRelSTEQ.source, simp add: indRelSTEQ.target)
    case (trans P Q R)
    assume \( P \sim [\cdot] < SRel, TRel > Q \) and \( Q \sim [\cdot] < SRel, TRel > R \)
    thus \( P \sim [\cdot] < SRel, TRel > R \)
    by (rule indRelSTEQ.trans)
  qed
next
  assume \( R \leq [\cdot] R < SRel, TRel > Q \)
  hence \( R \sim [\cdot] < SRel, TRel > Q \)
  proof (induct, simp add: indRelSTEQ.encL, simp add: indRelSTEQ.source, simp add: indRelSTEQ.target)
    case (trans P Q R)
    assume \( P \sim [\cdot] < SRel, TRel > Q \) and \( Q \sim [\cdot] < SRel, TRel > R \)
    thus \( P \sim [\cdot] < SRel, TRel > R \)
    by (rule indRelSTEQ.trans)
  qed

with \text{symmS \ symmT} show \( Q \sim [\cdot] < SRel, TRel > R \)
  using indRelSTEQ-symm[of SRel TRel]
  unfolding symcl-def
  by blast
qed

ultimately show \( P \sim [\cdot] < SRel, TRel > R \)
  using indRelSTEQ-symm
proof
  auto
  fix \( P \)
  assume \( P \sim [\cdot] < SRel, TRel > Q \)
  thus \((P, Q) \in (\text{symcl} (\text{indRelLSTPO} SRel TRel))^+\)
  proof
    induct
    case (encR S)
    show \((\text{SourceTerm} S, \text{TargetTerm} ([\cdot] S)) \in (\text{symcl} (\text{indRelLSTPO} SRel TRel))^+\)
      using indRelLSTPO.encL[of S SRel TRel]
      unfolding symcl-def
      by blast
  next
    case (encL S)
    show \((\text{TargetTerm} ([\cdot] S), \text{SourceTerm} S) \in (\text{symcl} (\text{indRelLSTPO} SRel TRel))^+\)
      using indRelLSTPO.encL[of S SRel TRel]
      unfolding symcl-def
      by blast
  next
    case (source S1 S2)
    assume \((S1, S2) \in SRel\)
    thus \((\text{SourceTerm} S1, \text{SourceTerm} S2) \in (\text{symcl} (\text{indRelLSTPO} SRel TRel))^+\)
using \texttt{indRelLSTPO.source}[of S1 S2 SRel TRel]

\textbf{unfolding} symcl-def

by blast

next case (target \(T1\) \(T2\))

assume \((T1, T2) \in TRel\)

thus \((\textnormal{Term} T1, \textnormal{Term} T2) \in (\textnormal{symcl} (\textnormal{indRelLSTPO} SRel TRel))^+\)

\textbf{unfolding} symcl-def

by blast

next

\textbf{case} (trans \(P\) \(Q\) \(R\))

\textbf{assume} \((P, Q) \in (\textnormal{symcl} (\textnormal{indRelLSTPO} SRel TRel))^+\)

\textbf{and} \((Q, R) \in (\textnormal{symcl} (\textnormal{indRelLSTPO} SRel TRel))^+\)

\textbf{thus} \((P, R) \in (\textnormal{symcl} (\textnormal{indRelLSTPO} SRel TRel))^+\)

by simp

qed

next

\textbf{fix} \(P\) \(Q\)

\textbf{assume} \((P, Q) \in (\textnormal{symcl} (\textnormal{indRelLSTPO} SRel TRel))^+\)

\textbf{thus} \(P \sim[\cdot]\langle SRel, TRel\rangle Q\)

\textbf{proof} \textbf{induct}

\textbf{fix} \(Q\)

\textbf{assume} \((P, Q) \in (\textnormal{symcl} (\textnormal{indRelLSTPO} SRel TRel))\)

\textbf{thus} \(P \sim[\cdot]\langle SRel, TRel\rangle Q\)

\textbf{proof} \textbf{auto}

\textbf{assume} \(P \sim[\cdot]\langle SRel, TRel\rangle Q\)

\textbf{thus} \(P \sim[\cdot]\langle SRel, TRel\rangle R\)

by (rule \texttt{indRelLSTPO.trans})

qed

next

\textbf{assume} \(Q \sim[\cdot]\langle SRel, TRel\rangle P\)

\textbf{hence} \(Q \sim[\cdot]\langle SRel, TRel\rangle P\)

\textbf{proof} \textbf{auto}

\textbf{assume} \(Q \sim[\cdot]\langle SRel, TRel\rangle P\)

\textbf{case} (trans \(P\) \(Q\) \(R\))

\textbf{assume} \(P \sim[\cdot]\langle SRel, TRel\rangle Q\) \textbf{and} \(Q \sim[\cdot]\langle SRel, TRel\rangle R\)

\textbf{thus} \(P \sim[\cdot]\langle SRel, TRel\rangle R\)

by (rule \texttt{indRelLSTPO.trans})

qed

with \texttt{symmS symmT show} \(P \sim[\cdot]\langle SRel, TRel\rangle Q\)

\textbf{using} \texttt{indRelLSTPO-symm}[of SRel TRel]

\textbf{unfolding} symcl-def

by blast

qed

next

\textbf{case} (step \(Q\) \(R\))

\textbf{assume} \(P \sim[\cdot]\langle SRel, TRel\rangle Q\)

\textbf{moreover assume} \((Q, R) \in (\textnormal{symcl} (\textnormal{indRelLSTPO} SRel TRel))\)

\textbf{hence} \(Q \sim[\cdot]\langle SRel, TRel\rangle R\)

\textbf{unfolding} symcl-def

\textbf{proof} \textbf{auto}

\textbf{assume} \(Q \sim[\cdot]\langle SRel, TRel\rangle R\)

\textbf{thus} \(Q \sim[\cdot]\langle SRel, TRel\rangle R\)

\textbf{proof} \textbf{auto}

\textbf{assume} \(Q \sim[\cdot]\langle SRel, TRel\rangle R\)

\textbf{thus} \(Q \sim[\cdot]\langle SRel, TRel\rangle R\)

\textbf{proof} \textbf{auto}

133
simp add: indRelSTEQ.target

case (trans P Q R)
  assume \(P \sim [\cdot] < SRel, TRel > Q\) and \(Q \sim [\cdot] < SRel, TRel > R\)
  thus \(P \sim [\cdot] < SRel, TRel > R\)
  by (rule indRelSTEQ.trans)
qed

next

assume \(R \leq [\cdot] L < SRel, TRel > Q\)
  hence \(R \sim [\cdot] < SRel, TRel > Q\)
proof (induct, simp add: indRelSTEQ.encL, simp add: indRelSTEQ.source, simp add: indRelSTEQ.target)
  case (trans P Q R)
  assume \(P \sim [\cdot] < SRel, TRel > Q\) and \(Q \sim [\cdot] < SRel, TRel > R\)
  thus \(P \sim [\cdot] < SRel, TRel > R\)
  by (rule indRelSTEQ.trans)
qed

with symmS symmT show \(Q \sim [\cdot] < SRel, TRel > R\)
  using indRelSTEQ-symm[of SRel TRel]
  unfolding sym-def
  by blast
qed

ultimately show \(P \sim [\cdot] < SRel, TRel > R\)
  by (rule indRelSTEQ.trans)
qed

If the relations indRelRST, indRelLST, or indRelST contain a pair of target terms, then this pair is also related by the considered target term relation. Similarly a pair of source terms is related by the considered source term relation.

lemma (in encoding) indRelRST-to-SRel:
  fixes SRel :: ('procS x 'procS) set
  and TRel :: ('procT x 'procT) set
  and SP SQ :: 'procS
  assumes rel: SourceTerm SP R<\[
  \cdot\]R < SRel, TRel > SourceTerm SQ
  shows \((SP, SQ) \in SRel\)
  using rel
  by (simp add: indRelRST.simps)

lemma (in encoding) indRelRST-to-TRel:
  fixes SRel :: ('procS x 'procS) set
  and TRel :: ('procT x 'procT) set
  and TP TQ :: 'procT
  assumes rel: TargetTerm TP R<\[
  \cdot\]R < SRel, TRel > TargetTerm TQ
  shows \((TP, TQ) \in TRel\)
  using rel
  by (simp add: indRelRST.simps)

lemma (in encoding) indRelLST-to-SRel:
  fixes SRel :: ('procS x 'procS) set
  and TRel :: ('procT x 'procT) set
  and SP SQ :: 'procS
  assumes rel: SourceTerm SP R<\[
  \cdot\]L < SRel, TRel > SourceTerm SQ
  shows \((SP, SQ) \in SRel\)
  using rel
  by (simp add: indRelLST.simps)

lemma (in encoding) indRelLST-to-TRel:
  fixes SRel :: ('procS x 'procS) set
  and TRel :: ('procT x 'procT) set
  and TP TQ :: 'procT
  assumes rel: TargetTerm TP R<\[
  \cdot\]L < SRel, TRel > TargetTerm TQ
  shows \((TP, TQ) \in TRel\)
  using rel
  by (simp add: indRelLST.simps)
assumes rel: TargetTerm TP R[\textbf{\textit{l}}<SRel,TRel>] TargetTerm TQ
shows (TP, TQ) ∈ TRel
  using rel
  by (simp add: indRelLST.simps)

lemma (in encoding) indRelST-to-SRel:
  fixes SRel :: (\textit{procS} × \textit{procS}) set
  and TRel :: (\textit{procT} × \textit{procT}) set
  and SP SQ :: \textit{procS}
  assumes rel: SourceTerm SP R[\textbf{\textit{l}}]<SRel,TRel> SourceTerm SQ
shows (SP, SQ) ∈ SRel
  using rel
  by (simp add: indRelST.simps)

lemma (in encoding) indRelST-to-TRel:
  fixes SRel :: (\textit{procS} × \textit{procS}) set
  and TRel :: (\textit{procT} × \textit{procT}) set
  and TP TQ :: \textit{procT}
  assumes rel: TargetTerm TP R[\textbf{\textit{l}}]<SRel,TRel> TargetTerm TQ
shows (TP, TQ) ∈ TRel
  using rel
  by (simp add: indRelST.simps)

If the relations indRelRSTPO or indRelLSTPO contain a pair of target terms, then this pair is also related by the transitive closure of the considered target term relation. Similarly a pair of source terms is related by the transitive closure of the source term relation. A pair of source and a target term results from the combination of pairs in the source relation, the target relation, and the encoding function. Note that, because of the symmetry, no similar condition holds for indRelSTEQ.

lemma (in encoding) indRelRSTPO-to-SRel-and-TRel:
  fixes SRel :: (\textit{procS} × \textit{procS}) set
  and TRel :: (\textit{procT} × \textit{procT}) set
  and P Q :: (\textit{procS}, \textit{procT}) Proc
  assumes P ≤[\textbf{\textit{r}}<SRel,TRel>] Q
shows ∀ SP SQ. SP ∈ S P ∧ SQ ∈ S Q → (SP, SQ) ∈ SRel
  and ∀ SP TQ. SP ∈ S P ∧ TQ ∈ T Q → (∃ S. (SP, S) ∈ SRel ∧ ([S], TQ) ∈ TRel*)
  and ∀ TP SQ. TP ∈ T P ∧ SQ ∈ S Q → False
  and ∀ TP TQ. TP ∈ T P ∧ TQ ∈ T Q → (TP, TQ) ∈ TRel*
  using assms
proof induct
  case (encR S)
  show ∀ SP SQ. SP ∈ S SourceTerm S ∧ SQ ∈ S TargetTerm ([S]) → (SP, SQ) ∈ SRel
    and ∀ TP SQ. TP ∈ T SourceTerm S ∧ SQ ∈ S TargetTerm ([S]) → False
    and ∀ TP TQ. TP ∈ T SourceTerm S ∧ TQ ∈ T TargetTerm ([S]) → (TP, TQ) ∈ TRel*
    by simp+
  have (S, S) ∈ SRel*
    by simp
  moreover have ([S], [S]) ∈ TRel*
    by simp
  ultimately show ∀ SP TQ. SP ∈ S SourceTerm S ∧ TQ ∈ T TargetTerm ([S]) →
    (∃ S. (SP, S) ∈ SRel* ∧ ([S], TQ) ∈ TRel*)
    by blast
next
case (source S1 S2)
assume (S1, S2) ∈ SRel
thus ∀ SP SQ. SP ∈ S SourceTerm S1 ∧ SQ ∈ S SourceTerm S2 → (SP, SQ) ∈ SRel*
  by simp
show ∀ SP TQ. SP ∈ S SourceTerm S1 ∧ TQ ∈ T SourceTerm S2 →
   (∃ S. (SP, S) ∈ SRel* ∧ ([S], TQ) ∈ TRel*)
  and ∀ TP SQ. TP ∈ T SourceTerm S1 ∧ SQ ∈ S SourceTerm S2 → False
  and ∀ TP TQ. TP ∈ T SourceTerm S1 ∧ TQ ∈ T SourceTerm S2 → (TP, TQ) ∈ TRel*
clarify

proof

next
case (target T1 T2)

show \( \forall SP SQ. SP \in S TargetTerm T1 \land SQ \in S TargetTerm T2 \rightarrow (SP, SQ) \in SRel^+ \)

and \( \forall SP TQ. SP \in S TargetTerm T1 \land TQ \in T TargetTerm T2 \rightarrow (\exists S. (SP, S) \in SRel^+ \land ([S], TQ) \in TRel^*) \)

and \( \forall TP SQ. TP \in T TargetTerm T1 \land SQ \in S TargetTerm T2 \rightarrow \text{False} \)

by simp

assume \( (T1, T2) \in TRel^+ \)

thus \( \forall TP TQ. TP \in T TargetTerm T1 \land TQ \in T TargetTerm T2 \rightarrow (TP, TQ) \in TRel^+ \)

by simp

next
case (trans P Q R)

assume A1: \( \forall SP SQ. SP \in S P \land SQ \in S Q \rightarrow (SP, SQ) \in SRel^+ \)

and A2: \( \forall SP TQ. SP \in S P \land TQ \in T Q \rightarrow (\exists S. (SP, S) \in SRel^+ \land ([S], TQ) \in TRel^*) \)

and A3: \( \forall TP SQ. TP \in T P \land SQ \in S Q \rightarrow \text{False} \)

and A4: \( \forall TP TQ. TP \in T P \land TQ \in T Q \rightarrow (TP, TQ) \in TRel^+ \)

and A5: \( \forall SQ SR. SQ \in S Q \land SR \in S R \rightarrow (SQ, SR) \in SRel^+ \)

and A6: \( \forall SQ TR. SQ \in S Q \land TR \in T R \rightarrow (\exists S. (SQ, S) \in SRel^+ \land ([S], TR) \in TRel^*) \)

and A7: \( \forall TQ SR. TQ \in T Q \land SR \in S R \rightarrow \text{False} \)

and A8: \( \forall TQ TR. TQ \in T Q \land TR \in T R \rightarrow (TQ, TR) \in TRel^+ \)

show \( \forall SP SR. SP \in S P \land SR \in S R \rightarrow (SP, SR) \in SRel^+ \)

proof clarify

fix SP SR

assume A9: \( SP \in S P \) and A10: \( SR \in S R \)

show \( (SP, SR) \in SRel^+ \)

proof (cases Q)

case (SourceTerm SQ)

assume A11: \( SQ \in S Q \)

with A1 A9 have \( (SP, SQ) \in SRel^+ \)

by simp

moreover from A5 A10 A11 have \( (SQ, SR) \in SRel^+ \)

by simp

ultimately show \( (SP, SR) \in SRel^+ \)

by simp

next
case (TargetTerm TQ)

assume TQ \( \in T Q \)

with A7 A10 show \( (SP, SR) \in SRel^+ \)

by blast

qed

qed

show \( \forall SP TR. SP \in S P \land TR \in T R \rightarrow (\exists S. (SP, S) \in SRel^+ \land ([S], TR) \in TRel^*) \)

proof clarify

fix SP TR

assume A9: \( SP \in S P \) and A10: \( TR \in T R \)

show \( \exists S. (SP, S) \in SRel^+ \land ([S], TR) \in TRel^* \)

proof (cases Q)

case (SourceTerm SQ)

assume A11: \( SQ \in S Q \)

with A6 A10 obtain \( S \) where A12: \( (SQ, S) \in SRel^+ \)

and A13: \( ([S], TR) \in TRel^* \)

by blast

from A1 A9 A11 have \( (SP, SQ) \in SRel^+ \)

by simp

from this A12 have \( (SP, S) \in SRel^+ \)

by simp

with A13 show \( \exists S. (SP, S) \in SRel^+ \land ([S], TR) \in TRel^* \)

by blast

next
\[
\begin{align*}
\text{case (TargetTerm } TQ) & \\
\text{assume } A11: TQ \in T Q & \\
\text{with } A2 A9 \text{ obtain } S \text{ where } A12: (SP, S) \in S\text{Rel}^* & \\
\text{and } A13: ([S], TQ) \in T\text{Rel}^* & \\
\text{by blast} & \\
\text{from } A8 A10 A11 \text{ have } (TQ, TR) \in T\text{Rel}^* & \\
\text{by simp} & \\
\text{with } A13 \text{ have } ([S], TR) \in T\text{Rel}^* & \\
\text{by simp} & \\
\text{with } A12 \text{ show } \exists S. (SP, S) \in S\text{Rel}^* \land ([S], TR) \in T\text{Rel}^* & \\
\text{by blast} & \\
\text{qed} & \\
\text{qed} & \\
\text{show } \forall TP \text{ SR. } TP \in T P \land \text{SR} \in S R \rightarrow False & \\
\text{proof clarify} & \\
\text{fix } TP \text{ SR} & \\
\text{assume } A9: TP \in T P \text{ and } A10: \text{SR} \in S R & \\
\text{show False} & \\
\text{proof (cases } Q) & \\
\text{case (SourceTerm } SQ) & \\
\text{assume } SQ \in S Q & \\
\text{with } A3 A9 \text{ show False} & \\
\text{by blast} & \\
\text{next} & \\
\text{case (TargetTerm } TQ) & \\
\text{assume } TQ \in T Q & \\
\text{with } A7 A10 \text{ show False} & \\
\text{by blast} & \\
\text{qed} & \\
\text{qed} & \\
\text{show } \forall TP \text{ TR. } TP \in T P \land TR \in T R \rightarrow (TP, TR) \in T\text{Rel}^+ & \\
\text{proof clarify} & \\
\text{fix } TP \text{ TR} & \\
\text{assume } A9: TP \in T P \text{ and } A10: TR \in T R & \\
\text{show } (TP, TR) \in T\text{Rel}^+ & \\
\text{proof (cases } Q) & \\
\text{case (SourceTerm } SQ) & \\
\text{assume } SQ \in S Q & \\
\text{with } A3 A9 \text{ show } (TP, TR) \in T\text{Rel}^+ & \\
\text{by blast} & \\
\text{next} & \\
\text{case (TargetTerm } TQ) & \\
\text{assume } A11: TQ \in T Q & \\
\text{with } A4 A9 \text{ have } (TP, TQ) \in T\text{Rel}^+ & \\
\text{by simp} & \\
\text{moreover from } A8 A10 A11 \text{ have } (TQ, TR) \in T\text{Rel}^+ & \\
\text{by simp} & \\
\text{ultimately show } (TP, TR) \in T\text{Rel}^+ & \\
\text{by simp} & \\
\text{qed} & \\
\text{qed} & \\
\text{qed} & \\
\text{lemma (in encoding) indRelLSTPO-to-SRel-and-TRel:} & \\
\text{fixes } S\text{Rel} :: (\text{procS} \times \text{proc}S) \text{ set} & \\
\text{and } T\text{Rel} :: (\text{procT} \times \text{proc}T) \text{ set} & \\
\text{and } P Q :: (\text{procS}, \text{proc}T) \text{ Proc} & \\
\text{assumes } P \leq \square L<\text{SRel}, T\text{Rel}>, Q & \\
\text{shows } \forall SP \text{ SQ. } SP \in S P \land \text{SQ} \in S Q \rightarrow (SP, SQ) \in S\text{Rel}^+ & \\
\text{and } \forall SP \text{ SQ. } SP \in S P \land TQ \in T Q \rightarrow False & \\
\text{and } \forall TP \text{ SQ. } TP \in T P \land \text{SQ} \in S Q \rightarrow (\exists S. (TP, [S]) \in T\text{Rel}^+ \land (S, SQ) \in S\text{Rel}^+) & \\
\end{align*}
\]
\[ \forall TP \ TQ. \ TP \in T \ P \land TQ \in T \ Q \rightarrow (TP, TQ) \in TRel^+ \]

**proof induct**

**case (encL S)**

**show** \( \forall SP SQ. \ SP \in S \ SourceTerm (\square[S]) \land SQ \in S \ SourceTerm S \rightarrow (SP, SQ) \in SRel^+ \)

and \( \forall SP TQ. \ SP \in S \ SourceTerm (\square[S]) \land TQ \in T \ SourceTerm S \rightarrow False \)

and \( \forall TP TQ. \ TP \in T \ SourceTerm (\square[S]) \land TQ \in T \ SourceTerm S \rightarrow (TP, TQ) \in TRel^+ \)

by **simp**

**have** \( (\square[S], \square[S]) \in TRel^+ \)

by **simp**

**moreover have** \( (S, S) \in SRel^+ \)

by **simp**

**ultimately show** \( \forall TP SQ. \ TP \in T \ SourceTerm (\square[S]) \land SQ \in S \ SourceTerm S \rightarrow (TP, [S]) \in TRel^+ \land (S, SQ) \in SRel^+ \)

by **blast**

**next**

**case (source S1 S2)**

**assume** \( (S1, S2) \in SRel \)

thus \( \forall SP SQ. \ SP \in S \ SourceTerm S1 \land SQ \in S \ SourceTerm S2 \rightarrow (SP, SQ) \in SRel^+ \)

by **simp**

**show** \( \forall SP TQ. \ SP \in S \ SourceTerm S1 \land TQ \in T \ SourceTerm S2 \rightarrow False \)

and \( \forall TP SQ. \ TP \in T \ SourceTerm S1 \land SQ \in S \ SourceTerm S2 \rightarrow (TP, [S]) \in TRel^+ \land (S, SQ) \in SRel^+ \)

and \( \forall TP TQ. \ TP \in T \ SourceTerm S1 \land TQ \in T \ SourceTerm S2 \rightarrow (TP, TQ) \in TRel^+ \)

by **simp**

**next**

**case (trans T1 T2)**

**show** \( \forall SP SQ. \ SP \in S \ SourceTerm T1 \land SQ \in S \ SourceTerm T2 \rightarrow (SP, SQ) \in SRel^+ \)

and \( \forall SP TQ. \ SP \in S \ SourceTerm T1 \land TQ \in T \ SourceTerm T2 \rightarrow False \)

and \( \forall TP SQ. \ TP \in T \ SourceTerm T1 \land SQ \in S \ SourceTerm T2 \rightarrow (TP, [S]) \in TRel^+ \land (S, SQ) \in SRel^+ \)

by **simp**

**assume** \( (T1, T2) \in TRel \)

thus \( \forall TP TQ. \ TP \in T \ SourceTerm T1 \land TQ \in T \ SourceTerm T2 \rightarrow (TP, TQ) \in TRel^+ \)

by **simp**

**next**

**case (sourceTerm SQ)**

**assume** \( A1: \forall SP SQ. \ SP \in S \ P \land SQ \in S \ Q \rightarrow (SP, SQ) \in SRel^+ \)

and \( A2: \forall SP TQ. \ SP \in S \ P \land TQ \in T \ Q \rightarrow False \)

and \( A3: \forall TP SQ. \ TP \in T \ P \land SQ \in S \ Q \rightarrow (TP, [S]) \in TRel^+ \land (S, SQ) \in SRel^+ \)

and \( A4: \forall TP TQ. \ TP \in T \ P \land TQ \in T \ Q \rightarrow (TP, TQ) \in TRel^+ \)

and \( A5: \forall SQ SR. \ SQ \in S \ Q \land SR \in S \ R \rightarrow (SQ, SR) \in SRel^+ \)

and \( A6: \forall SQ TR. \ SQ \in S \ Q \land TR \in T \ R \rightarrow False \)

and \( A7: \forall TP SQ. \ TP \in T \ Q \land SR \in S \ R \rightarrow (TP, [S]) \in TRel^+ \land (S, SR) \in SRel^+ \)

and \( A8: \forall TP TR. \ TP \in T \ Q \land TR \in T \ R \rightarrow (TP, [S]) \in TRel^+ \land (S, SR) \in SRel^+ \)

**show** \( \forall SP SR. \ SP \in S \ P \land SR \in S \ R \rightarrow (SP, SR) \in SRel^+ \)

**proof clarify**

**fix** \( SP SR \)

**assume** \( A9: \ SP \in S \ P \land A10: \ SR \in S \ R \)

**show** \( (SP, SR) \in SRel^+ \)

**proof (cases Q)**

**case (SourceTerm SQ)**

**assume** \( A11: \ SQ \in S \ Q \)

with \( A1 A9 \ have \ (SP, SQ) \in SRel^+ \)

by **simp**

**moreover from A5 A10 A11 have** \( (SQ, SR) \in SRel^+ \)

by **simp**

ultimately show \( (SP, SR) \in SRel^+ \)

by **simp**
case (TargetTerm TQ)
assume TQ ∈ T Q
with A2 A9 show (SP, SR) ∈ SRel+
  by blast
qed
qed

show ∀ SP TR. SP ∈ S P ∧ TR ∈ T R → False
proof clarify
  fix SP TR
  assume A9: SP ∈ S P and A10: TR ∈ T R
  show False
  proof (cases Q)
    case (SourceTerm SQ)
    assume SQ ∈ S Q
    with A6 A10 show False
      by blast
  next
  case (TargetTerm TQ)
  assume TQ ∈ T Q
  with A2 A9 show False
    by blast
qed
qed

show ∀ TP SR. TP ∈ T P ∧ SR ∈ S R → (∃ S. (TP, [S]) ∈ TRel* ∧ (S, SR) ∈ SRel*)
proof clarify
  fix TP SR
  assume A9: TP ∈ T P and A10: SR ∈ S R
  show ∃ S. (TP, [S]) ∈ TRel* ∧ (S, SR) ∈ SRel*
  proof (cases Q)
    case (SourceTerm SQ)
    assume A11: SQ ∈ S Q
    with A3 A9 obtain S where A12: (TP, [S]) ∈ TRel* and A13: (S, SQ) ∈ SRel*
      by blast
    from A5 A10 A11 have (SQ, SR) ∈ SRel*
      by simp
    with A13 have (S, SR) ∈ SRel*
      by simp
    with A12 show ∃ S. (TP, [S]) ∈ TRel* ∧ (S, SR) ∈ SRel*
      by blast
  next
  case (TargetTerm TQ)
  assume A11: TQ ∈ T Q
  with A7 A10 obtain S where A12: (TQ, [S]) ∈ TRel* and A13: (S, SR) ∈ SRel*
    by blast
  from A4 A9 A11 have (TP, TQ) ∈ TRel*
    by simp
  from this A12 have (TP, [S]) ∈ TRel*
    by simp
  with A13 show ∃ S. (TP, [S]) ∈ TRel* ∧ (S, SR) ∈ SRel*
    by blast
qed
qed

show ∀ TP TR. TP ∈ T P ∧ TR ∈ T R → (TP, TR) ∈ TRel+
proof clarify
  fix TP TR
  assume A9: TP ∈ T P and A10: TR ∈ T R
  show (TP, TR) ∈ TRel+
  proof (cases Q)
    case (SourceTerm SQ)
    assume SQ ∈ S Q
    with A6 A10 show (TP, TR) ∈ TRel+
  next
  case (TargetTerm TQ)
  assume TQ ∈ T Q
  with A2 A9 show (TP, TR) ∈ TRel+
  by blast
qed

by blast

next
case (TargetTerm TQ)
assume A11: TQ ∈ TQ
with A4 A9 have (TP, TQ) ∈ TRel+
  by simp
moreover from A8 A10 A11 have (TQ, TR) ∈ TRel+
  by simp
ultimately show (TP, TR) ∈ TRel+
  by simp
qed
qed
qed

If indRelRSTPO, indRelLSTPO, or indRelSTPO preserves barbs then so do the corresponding source term and target term relations.

lemma (in encoding-wrt-barbs) rel-with-source-impl-SRel-preserves-barbs:
fixes SRel :: ('procS × 'procS) set
and Rel :: ('procS, 'procT) Proc × ('procS, 'procT) Proc set
assumes preservation: rel-preserves-barbs Rel (STCalWB SWB TWB)
and sourceInRel: ∀ S1 S2. (S1, S2) ∈ SRel → (SourceTerm S1, SourceTerm S2) ∈ Rel
shows rel-preserves-barbs SRel SWB
proof clarify
fix SP SQ a
assume (SP, SQ) ∈ SRel
with sourceInRel have (SourceTerm SP, SourceTerm SQ) ∈ Rel
  by blast
moreover assume SP ↓ <SWB> a
hence SourceTerm SP ↓ a
  by simp
ultimately have SourceTerm SQ ↓ a
  using preservation preservation-of-barbs-in-barbed-encoding[where Rel=Rel]
  by blast
thus SQ ↓ <SWB> a
  by simp
qed

lemma (in encoding-wrt-barbs) indRelRSTPO-impl-SRel-and-TRel-preserve-barbs:
fixes SRel :: ('procS × 'procS) set
and TRel :: ('procT × 'procT) set
assumes preservation: rel-preserves-barbs (indRelRSTPO SRel TRel) (STCalWB SWB TWB)
shows rel-preserves-barbs SRel SWB
and rel-preserves-barbs TRel TWB
proof
  show rel-preserves-barbs SRel SWB
    using preservation rel-with-source-impl-SRel-preserves-barbs[where
    Rel=indRelRSTPO SRel TRel and SRel=SRel]
    by (simp add: indRelRSTPO.source)
next
  show rel-preserves-barbs TRel TWB
    using preservation rel-with-target-impl-TRel-preserves-barbs[where
    Rel=indRelRSTPO SRel TRel and TRel=TRel]
    by (simp add: indRelRSTPO.target)
qed

lemma (in encoding-wrt-barbs) indRelLSTPO-impl-SRel-and-TRel-preserve-barbs:
fixes SRel :: ('procS × 'procS) set
and TRel :: ('procT × 'procT) set
assumes preservation: rel-preserves-barbs (indRelLSTPO SRel TRel) (STCalWB SWB TWB)
shows rel-preserves-barbs SRel SWB
and rel-preserves-barbs TRel TWB

proof –

show rel-preserves-barbs SRel SWB
  using preservation rel-with-source-impl-SRel-preserves-barbs
   where Rel=indRlLSTPO SRel TRel and SRel=SRel
   by (simp add: indRlLSTPO.source)

next

show rel-preserves-barbs TRel TWB
  using preservation rel-with-target-impl-TRel-preserves-barbs
   where Rel=indRlLSTPO SRel TRel and TRel=TRel
   by (simp add: indRlLSTPO.target)

qed

lemma (in encoding-wrt-barbs) indRlSTEQ-impl-SRel-and-TRel-preserve-barbs:
  fixes SRel :: ('procS × 'procS) set
  and TRel :: ('procT × 'procT) set
  assumes preservation: rel-preserves-barbs (indRlSTEQ SRel TRel) (STCalWB SWB TWB)
  shows rel-preserves-barbs SRel SWB
  and rel-preserves-barbs TRel TWB
proof
  clarify
  fix SP SQ a SP'
  assume (SP, SQ) ∈ SRel
  with sourceInRel have (SourceTerm SP, SourceTerm SQ) ∈ Rel
    by blast
  moreover assume SP |-- (Calculus SWB)* SP' and SP'↓<SWB>a
  hence SourceTerm SP↓.a
    by blast
  ultimately have SourceTerm SQ↓.a
    using preservation weak-preservation-of-barbs-in-barbed-encoding
    where Rel=Rel
    by blast
  thus SQ↓<SWB>a
    by simp

qed

lemma (in encoding-wrt-barbs) indRlRSTPO-impl-SRel-and-TRel-weakly-preserve-barbs:
  fixes SRel :: ('procS × 'procS) set
  and TRel :: ('procT × 'procT) set
  assumes preservation: rel-weakly-preserves-barbs (indRlRSTPO SRel TRel) (STCalWB SWB TWB)
  and sourceInRel: ∀ S1 S2. (S1, S2) ∈ SRel −→ (SourceTerm S1, SourceTerm S2) ∈ Rel
  shows rel-weakly-preserves-barbs SRel SWB
  and rel-weakly-preserves-barbs TRel TWB
proof
  clarify
  fix SP SQ a SP'
  assume (SP, SQ) ∈ SRel
  with sourceInRel have (SourceTerm SP, SourceTerm SQ) ∈ Rel
    by blast
  moreover assume SP |-- (Calculus SWB)* SP' and SP'↓<SWB>a
  hence SourceTerm SP↓.a
    by blast
  ultimately have SourceTerm SQ↓.a
    using preservation weak-preservation-of-barbs-in-barbed-encoding
    where Rel=Rel
    by blast
  thus SQ↓<SWB>a
    by simp

qed
\[ \text{Rel} = \text{indRelRSTPO} \text{ SRel TRel and SRel} = \text{SRel} \]

by (simp add: \text{indRelRSTPO.source})

next

show rel-weakly-preserves-barbs \text{TRel TWB}

using \text{preservation rel-with-target-impl-TRel-weakly-preserves-barbs} 

\begin{align*}
\text{Rel} &= \text{indRelRSTPO} \text{ SRel TRel and TRel} = \text{TRel} \\
\end{align*}

by (simp add: \text{indRelRSTPO.target})

qed

lemma (in encoding-wrt-barbs) \text{indRelLSTPO-impl-SRel-and-TRel-weakly-preserve-barbs}:

fixes \text{SRel} :: (\text{'procS} × \text{'procS}) set

and \text{TRel} :: (\text{'procT} × \text{'procT}) set

assumes \text{preservation: rel-weakly-preserves-barbs} (\text{indRelLSTPO} \text{ SRel TRel}) (\text{STCalWB SWB TWB})

shows rel-weakly-preserves-barbs \text{SRel SWB}

and rel-weakly-preserves-barbs \text{TRel TWB}

proof –

show rel-weakly-preserves-barbs \text{SRel SWB}

using \text{preservation rel-with-source-impl-SRel-weakly-preserves-barbs} 

\begin{align*}
\text{Rel} &= \text{indRelLSTPO} \text{ SRel TRel and SRel} = \text{SRel} \\
\end{align*}

by (simp add: \text{indRelLSTPO.source})

next

show rel-weakly-preserves-barbs \text{TRel TWB}

using \text{preservation rel-with-target-impl-TRel-weakly-preserves-barbs} 

\begin{align*}
\text{Rel} &= \text{indRelLSTPO} \text{ SRel TRel and TRel} = \text{TRel} \\
\end{align*}

by (simp add: \text{indRelLSTPO.target})

qed

lemma (in encoding-wrt-barbs) \text{indRelSTEQ-impl-SRel-and-TRel-weakly-preserve-barbs}:

fixes \text{SRel} :: (\text{'procS} × \text{'procS}) set

and \text{TRel} :: (\text{'procS} × \text{'procS}) set

assumes \text{preservation: rel-weakly-preserves-barbs} (\text{indRelSTEQ} \text{ SRel TRel}) (\text{STCalWB SWB TWB})

shows rel-weakly-preserves-barbs \text{SRel SWB}

and rel-weakly-preserves-barbs \text{TRel TWB}

proof –

show rel-weakly-preserves-barbs \text{SRel SWB}

using \text{preservation rel-with-source-impl-SRel-weakly-preserves-barbs} 

\begin{align*}
\text{Rel} &= \text{indRelSTEQ} \text{ SRel TRel and SRel} = \text{SRel} \\
\end{align*}

by (simp add: \text{indRelSTEQ.source})

next

show rel-weakly-preserves-barbs \text{TRel TWB}

using \text{preservation rel-with-target-impl-TRel-weakly-preserves-barbs} 

\begin{align*}
\text{Rel} &= \text{indRelSTEQ} \text{ SRel TRel and TRel} = \text{TRel} \\
\end{align*}

by (simp add: \text{indRelSTEQ.target})

qed

If \text{indRelRSTPO, indRelLSTPO, or indRelSTPO} reflects barbs then so do the corresponding source term and target term relations.

**lemma (in encoding-wrt-barbs) rel-with-source-impl-SRel-reflects-barbs:**

fixes \text{SRel} :: (\text{'procS} × \text{'procS}) set

and \text{Rel} :: (\text{'procS} × \text{'procS}) set

assumes reflection: rel-reflects-barbs \text{Rel} (\text{STCalWB SWB TWB})

and sourceInRel: \( \forall S1 S2. (S1, S2) ∈ \text{SRel} \rightarrow (\text{SourceTerm} S1, \text{SourceTerm} S2) ∈ \text{Rel} \)

shows rel-reflects-barbs \text{SRel SWB}

proof clarify

fix \text{SP SQ a}

assume \text{(SP, SQ) ∈ SRel}

with sourceInRel have \( (\text{SourceTerm} \text{SP, SourceTerm} \text{SQ}) ∈ \text{Rel} \)

by blast

moreover assume \text{SQ, <SWB> a}

hence \text{SourceTerm} \text{SQ, a}

142
ultimately have \( \text{SourceTerm } \text{SP}_a \)
by \text{simp}

thus \( \text{SP}_a \langle \text{SWB} \rangle \)
by \text{simp}

qed

\text{lemma (in encoding-wrt-barbs) indRelRSTPO-impl-SRel-and-TRel-reflect-barbs:}

\begin{align*}
\text{fixes } SRel &:: (\text{procS} \times \text{procS}) \text{ set} \\
\text{and } TRel &:: (\text{procT} \times \text{procT}) \text{ set} \\
\text{assumes reflection: rel-reflects-barbs (indRelRSTPO SRel TRel) (STCalWB SWB TWB)} \\
\text{shows rel-reflects-barbs SRel SWB} \\
\text{and rel-reflects-barbs TRel TWB}
\end{align*}

\text{proof –}

show rel-reflects-barbs SRel SWB
\begin{align*}
\text{using reflection rel-with-source-impl-SRel-reflects-barbs[where} \\
\text{Rel=} \text{indRelRSTPO SRel TRel and SRel=} SRel
\end{align*}
by (\text{simp add: indRelRSTPO.source})

next

show rel-reflects-barbs TRel TWB
\begin{align*}
\text{using reflection rel-with-target-impl-TRel-reflects-barbs[where} \\
\text{Rel=} \text{indRelRSTPO SRel TRel and TRel=} TRel
\end{align*}
by (\text{simp add: indRelRSTPO.target})

qed

\text{lemma (in encoding-wrt-barbs) indRelLSTPO-impl-SRel-and-TRel-reflect-barbs:}

\begin{align*}
\text{fixes } SRel &:: (\text{procS} \times \text{procS}) \text{ set} \\
\text{and } TRel &:: (\text{procS} \times \text{procS}) \text{ set} \\
\text{assumes reflection: rel-reflects-barbs (indRelLSTPO SRel TRel) (STCalWB SWB TWB)} \\
\text{shows rel-reflects-barbs SRel SWB} \\
\text{and rel-reflects-barbs TRel TWB}
\end{align*}

\text{proof –}

show rel-reflects-barbs SRel SWB
\begin{align*}
\text{using reflection rel-with-source-impl-SRel-reflects-barbs[where} \\
\text{Rel=} \text{indRelLSTPO SRel TRel and SRel=} SRel
\end{align*}
by (\text{simp add: indRelLSTPO.source})

next

show rel-reflects-barbs TRel TWB
\begin{align*}
\text{using reflection rel-with-target-impl-TRel-reflects-barbs[where} \\
\text{Rel=} \text{indRelLSTPO SRel TRel and TRel=} TRel
\end{align*}
by (\text{simp add: indRelLSTPO.target})

qed

\text{lemma (in encoding-wrt-barbs) indRelSTEQ-impl-SRel-and-TRel-reflect-barbs:}

\begin{align*}
\text{fixes } SRel &:: (\text{procS} \times \text{procS}) \text{ set} \\
\text{and } TRel &:: (\text{procS} \times \text{procS}) \text{ set} \\
\text{assumes reflection: rel-reflects-barbs (indRelSTEQ SRel TRel) (STCalWB SWB TWB)} \\
\text{shows rel-reflects-barbs SRel SWB} \\
\text{and rel-reflects-barbs TRel TWB}
\end{align*}

\text{proof –}

show rel-reflects-barbs SRel SWB
\begin{align*}
\text{using reflection rel-with-source-impl-SRel-reflects-barbs[where} \\
\text{Rel=} \text{indRelSTEQ SRel TRel and SRel=} SRel
\end{align*}
by (\text{simp add: indRelSTEQ.source})

next

show rel-reflects-barbs TRel TWB
\begin{align*}
\text{using reflection rel-with-target-impl-TRel-reflects-barbs[where} \\
\text{Rel=} \text{indRelSTEQ SRel TRel and TRel=} TRel
\end{align*}
by (\text{simp add: indRelSTEQ.target})

qed
lemma (in encoding-wrt-barbs) rel-with-source-impl-SRel-weakly-reflects-barbs:
  fixes SRel :: 
  \emph{\('\text{procS} \times \text{procS}'\)} set
  \and Rel :: 
  \emph{\('\text{procS}, \text{procT}'\) Proc \times \('\text{procS}, \text{procT}'\) Proc} set
  \assumes reflection: rel-weakly-reflects-barbs Rel (STCalWB SWB TWB)
  \and sourceInRel: \forall S1 S2. (S1, S2) \in SRel \rightarrow (\text{SourceTerm S1}, \text{SourceTerm S2}) \in Rel
  \shows rel-weakly-reflects-barbs SRel SWB

proof clarify
  fix SP SQ a SQ'
  assume (SP, SQ) \in SRel
  with sourceInRel have (SourceTerm SP, SourceTerm SQ) \in Rel
  by blast
  moreover assume SQ \rightarrow\rightarrow (Calculus SWB)* SQ' and SQ' \downarrow<SWB>a
  hence SourceTerm SQ\downarrow.a
  by blast
  ultimately have SourceTerm SP\downarrow.a
  using reflection weak-reflection-of-barbs-in-barbed-encoding[where Rel=Rel]
  by blast
  thus SP\downarrow<SWB>a
  by simp

qed

lemma (in encoding-wrt-barbs) indRelRSTPO-impl-SRel-and-TRel-weakly-reflect-barbs:
  fixes SRel :: 
  \emph{\('\text{procS} \times \text{procS}'\)} set
  \and TRel :: 
  \emph{\('\text{procT} \times \text{procT}'\)} set
  \assumes reflection: rel-weakly-reflects-barbs (indRelRSTPO SRel TRel) (STCalWB SWB TWB)
  \shows rel-weakly-reflects-barbs SRel SWB
  \and rel-weakly-reflects-barbs TRel TWB

proof
  show rel-weakly-reflects-barbs SRel SWB
    using reflection rel-with-source-impl-SRel-weakly-reflects-barbs[where
    Rel=indRelRSTPO SRel TRel and SRel=SRel]
    by (simp add: indRelRSTPO.source)
  next
  show rel-weakly-reflects-barbs TRel TWB
    using reflection rel-with-target-impl-TRel-weakly-reflects-barbs[where
    Rel=indRelRSTPO SRel TRel and TRel=TRel]
    by (simp add: indRelRSTPO.target)

qed

lemma (in encoding-wrt-barbs) indRelLSTPO-impl-SRel-and-TRel-weakly-reflect-barbs:
  fixes SRel :: 
  \emph{\('\text{procS} \times \text{procS}'\)} set
  \and TRel :: 
  \emph{\('\text{procT} \times \text{procT}'\)} set
  \assumes reflection: rel-weakly-reflects-barbs (indRelLSTPO SRel TRel) (STCalWB SWB TWB)
  \shows rel-weakly-reflects-barbs SRel SWB
  \and rel-weakly-reflects-barbs TRel TWB

proof
  show rel-weakly-reflects-barbs SRel SWB
    using reflection rel-with-source-impl-SRel-weakly-reflects-barbs[where
    Rel=indRelLSTPO SRel TRel and SRel=SRel]
    by (simp add: indRelLSTPO.source)
  next
  show rel-weakly-reflects-barbs TRel TWB
    using reflection rel-with-target-impl-TRel-weakly-reflects-barbs[where
    Rel=indRelLSTPO SRel TRel and TRel=TRel]
    by (simp add: indRelLSTPO.target)

qed

lemma (in encoding-wrt-barbs) indRelSTEQ-impl-SRel-and-TRel-weakly-reflect-barbs:
  fixes SRel :: 
  \emph{\('\text{procS} \times \text{procS}'\)} set
  \and TRel :: 
  \emph{\('\text{procT} \times \text{procT}'\)} set
assumes reflection: rel-weakly-reflects-barbs (indRelSTEQ SRel TRel) (STCalWB SWB TWB)
shows rel-weakly-reflects-barbs SRel SWB
and rel-weakly-reflects-barbs TRel TWB
proof –
  show rel-weakly-reflects-barbs SRel SWB
    using reflection rel-with-source-impl-SRel-weakly-reflects-barbs
      where
      Rel=indRelSTEQ SRel TRel and SRel=SRel
    by (simp add: indRelSTEQ source)
next
  show rel-weakly-reflects-barbs TRel TWB
    using reflection rel-with-target-impl-TRel-weakly-reflects-barbs
      where
      Rel=indRelSTEQ SRel TRel and TRel=TRel
    by (simp add: indRelSTEQ target)
qed

If indRelRSTPO, indRelLSTPO, or indRelSTPO respects barbs then so do the corresponding source term and target term relations.

lemma (in encoding-wrt-barbs) indRelRSTPO-impl-SRel-and-TRel-respect-barbs:
  fixes SRel :: (′procS × ′procS) set
and TRel :: (′procT × ′procT) set
assumes respection: rel-respects-barbs (indRelRSTPO SRel TRel) (STCalWB SWB TWB)
shows rel-respects-barbs SRel SWB
and rel-respects-barbs TRel TWB
proof –
  show rel-respects-barbs SRel SWB
    using respection
      indRelRSTPO-impl-SRel-and-TRel-preserve-barbs(1)
      where SRel=SRel and TRel=TRel
    by blast
next
  show rel-respects-barbs TRel TWB
    using respection
      indRelRSTPO-impl-SRel-and-TRel-preserve-barbs(2)
      where SRel=SRel and TRel=TRel
    by blast
qed

lemma (in encoding-wrt-barbs) indRelLSTPO-impl-SRel-and-TRel-respect-barbs:
  fixes SRel :: (′procS × ′procS) set
and TRel :: (′procT × ′procT) set
assumes respection: rel-respects-barbs (indRelLSTPO SRel TRel) (STCalWB SWB TWB)
shows rel-respects-barbs SRel SWB
and rel-respects-barbs TRel TWB
proof –
  show rel-respects-barbs SRel SWB
    using respection
      indRelLSTPO-impl-SRel-and-TRel-preserve-barbs(1)
      where SRel=SRel and TRel=TRel
    by blast
next
  show rel-respects-barbs TRel TWB
    using respection
      indRelLSTPO-impl-SRel-and-TRel-preserve-barbs(2)
      where SRel=SRel and TRel=TRel
    by blast
qed

lemma (in encoding-wrt-barbs) indRelSTEQ-impl-SRel-and-TRel-respect-barbs:
  fixes SRel :: (′procS × ′procS) set
and TRel :: (′procT × ′procT) set

assumes respection: rel-respects-barbs (indRelSTEQ SRel TRel) (STCalWB SWB TWB)
shows rel-respects-barbs SRel SWB
    and rel-respects-barbs TRel TWB
proof –
    show rel-respects-barbs SRel SWB
        using respection
        indRelSTEQ-impl-SRel-and-TRel-preserve-barbs(1)[where SRel=SRel and TRel=TRel]
        indRelSTEQ-impl-SRel-and-TRel-reflect-barbs(1)[where SRel=SRel and TRel=TRel]
        by blast
next
    show rel-respects-barbs TRel TWB
        using respection
        indRelSTEQ-impl-SRel-and-TRel-preserve-barbs(2)[where SRel=SRel and TRel=TRel]
        indRelSTEQ-impl-SRel-and-TRel-reflect-barbs(2)[where SRel=SRel and TRel=TRel]
        by blast
qed

lemma (in encoding-wrt-barbs) indRelRSTPO-impl-SRel-and-TRel-weakly-respect-barbs:
fixes SRel :: ('procS × 'procS) set
    and TRel :: ('procT × 'procT) set
assumes respection: rel-weakly-respects-barbs (indRelRSTPO SRel TRel) (STCalWB SWB TWB)
shows rel-weakly-respects-barbs SRel SWB
    and rel-weakly-respects-barbs TRel TWB
proof –
    show rel-weakly-respects-barbs SRel SWB
        using respection
        indRelRSTPO-impl-SRel-and-TRel-weakly-preserve-barbs(1)[where SRel=SRel
                        and TRel=TRel]
        indRelRSTPO-impl-SRel-and-TRel-weakly-reflect-barbs(1)[where SRel=SRel
                        and TRel=TRel]
        by blast
next
    show rel-weakly-respects-barbs TRel TWB
        using respection
        indRelRSTPO-impl-SRel-and-TRel-weakly-preserve-barbs(2)[where SRel=SRel
                        and TRel=TRel]
        indRelRSTPO-impl-SRel-and-TRel-weakly-reflect-barbs(2)[where SRel=SRel
                        and TRel=TRel]
        by blast
qed

lemma (in encoding-wrt-barbs) indRelLSTPO-impl-SRel-and-TRel-weakly-respect-barbs:
fixes SRel :: ('procS × 'procS) set
    and TRel :: ('procT × 'procT) set
assumes respection: rel-weakly-respects-barbs (indRelLSTPO SRel TRel) (STCalWB SWB TWB)
shows rel-weakly-respects-barbs SRel SWB
    and rel-weakly-respects-barbs TRel TWB
proof –
    show rel-weakly-respects-barbs SRel SWB
        using respection
        indRelLSTPO-impl-SRel-and-TRel-weakly-preserve-barbs(1)[where SRel=SRel
                        and TRel=TRel]
        indRelLSTPO-impl-SRel-and-TRel-weakly-reflect-barbs(1)[where SRel=SRel
                        and TRel=TRel]
        by blast
next
    show rel-weakly-respects-barbs TRel TWB
        using respection
        indRelLSTPO-impl-SRel-and-TRel-weakly-preserve-barbs(2)[where SRel=SRel
                        and TRel=TRel]
        indRelLSTPO-impl-SRel-and-TRel-weakly-reflect-barbs(2)[where SRel=SRel
                        and TRel=TRel]
        by blast
qed
lemma (in encoding-wrt-barbs) indRelSTEQ-impl-SRel-and-TRel-weakly-respect-barbs:
  fixes SRel :: ('procS × 'procS) set
  and TRel :: ('procT × 'procT) set
  assumes respection: rel-weakly-respects-barbs (indRelSTEQ SRel TRel) (STCalWB SWB TWB)
  shows rel-weakly-respects-barbs SRel SWB
  and rel-weakly-respects-barbs TRel TWB
proof
  show rel-weakly-respects-barbs SRel SWB
    using respection indRelSTEQ-impl-SRel-and-TRel-weakly-preserve-barbs(1)[where SRel=SRel
    and TRel=TRel]
    by blast
next
  show rel-weakly-respects-barbs TRel TWB
    using respection indRelSTEQ-impl-SRel-and-TRel-weakly-preserve-barbs(2)[where SRel=SRel
    and TRel=TRel]
    by blast
qed

If TRel is reflexive then ind relRTPO is a subrelation of indRelTEQ. If SRel is reflexive then indRelRTPO is a subrelation of indRelRTPO. Moreover, indRelRSTPO is a subrelation of indRelSTEQ.

lemma (in encoding) indRelRTPO-to-indRelTEQ:
  fixes TRel :: ('procT × 'procT) set
  and P Q :: ('procS,'procT) Proc
  assumes rel: P ⊲ R<TRel> Q
  and reflT: refl TRel
  shows P ∼ R<TRel> Q
using rel
proof induct
  case (encR S)
  show SourceTerm S ∼ T<TRel> TargetTerm (\[S\])
    by (rule indRelTEQ.encR)
next
  case (source S)
  from reflT show SourceTerm S ∼ T<TRel> SourceTerm S
    using indRelTEQ-refl[of TRel]
    unfolding refl-on-def
    by simp
next
  case (target T1 T2)
  assume (T1, T2) ∈ TRel
  thus TargetTerm T1 ∼ T<TRel> TargetTerm T2
    by (rule indRelTEQ.target)
next
  case (trans TP TQ TR)
  assume TP ∼ T<TRel> TQ and TQ ∼ T<TRel> TR
  thus TP ∼ T<TRel> TR
    by (rule indRelTEQ.trans)
qed

lemma (in encoding) indRelRTPO-to-indRelRSTPO:
  fixes SRel :: ('procS × 'procS) set
  and TRel :: ('procT × 'procT) set
  and P Q :: ('procS,'procT) Proc
  assumes rel: P ⊲ R<TRel> Q
  and reflS: refl SRel
  shows P ⊲ R<SRel,TRel> Q
using rel
proof induct
  case (encR S)
  show SourceTerm S \leq SRel, TRel> TargetTerm ([S])
    by (rule indRelRSTPO.encR)
next
  case (source S)
  from reflS show SourceTerm S \leq SRel, TRel> SourceTerm S
    unfolding refl-on-def
    by (simp add: indRelRSTPO.source)
next
  case (target T1 T2)
  assume (T1, T2) \in TRel
  thus TargetTerm T1 \leq SRel, TRel> TargetTerm T2
    by (rule indRelRSTPO.target)
next
  case (trans P Q R)
  assume P \leq SRel, TRel> Q and Q \leq SRel, TRel> R
  thus P \leq SRel, TRel> R
    by (rule indRelRSTPO.trans)
qed

lemma (in encoding) indRelRSTPO-to-indRelSTEQ:
  fixes SRel :: ('procS \times 'procS) set
  and TRel :: ('procT \times 'procT) set
  and P Q :: ('procS, 'procT) Proc
  assumes rel: P \leq SRel, TRel> Q
  shows P \lhd SRel, TRel> Q
  using rel
proof induct
  case (encR S)
  show SourceTerm S \lhd SRel, TRel> TargetTerm ([S])
    by (rule indRelSTEQ.encR)
next
  case (source S1 S2)
  assume (S1, S2) \in SRel
  thus SourceTerm S1 \lhd SRel, TRel> SourceTerm S2
    by (rule indRelSTEQ.source)
next
  case (target T1 T2)
  assume (T1, T2) \in TRel
  thus TargetTerm T1 \lhd SRel, TRel> TargetTerm T2
    by (rule indRelSTEQ.target)
next
  case (trans P Q R)
  assume P \lhd SRel, TRel> Q and Q \lhd SRel, TRel> R
  thus P \lhd SRel, TRel> R
    by (rule indRelSTEQ.trans)
qed

If indRelRTPO is a bisimulation and SRel is a reflexive bisimulation then also indRelRSTPO is a bisimulation.

lemma (in encoding) indRelRTPO-weak-reduction-bisimulation-impl-indRelRSTPO-bisimulation:
  fixes SRel :: ('procS \times 'procS) set
  and TRel :: ('procT \times 'procT) set
  assumes bisimT: weak-reduction-bisimulation (indRelRTPO TRel) (STCal Source Target)
  and bisimS: weak-reduction-bisimulation SRel Source
  and reflS: refl SRel
  shows weak-reduction-bisimulation (indRelRSTPO SRel TRel) (STCal Source Target)
proof auto
fix \( P, Q, P' \)

\[ \text{assume } \exists Q' . \ Q \mapsto_{\parallel R} (\text{STCal Source Target})* \ P' \]

\[ \text{thus } \exists Q' . \ Q \mapsto_{\parallel R} (\text{STCal Source Target})* \ Q' \land P' \leq_{\parallel R} \] \( \exists Q' \)

\[ \text{proof (induct arbitrary: } P' \) \]

\[ \text{case (encS } R) \]

\[ \text{have } \text{SourceTerm } S \leq_{\parallel RT} \text{TargetTerm } ([S]) \]

\[ \begin{array}{l}
  \text{by (rule indRelRTPO.encR)} \\
  \text{moreover assume } \text{SourceTerm } S \mapsto_{\parallel (\text{STCal Source Target})*} \ P' \\
  \text{ultimately obtain } Q' \text{ where } A1 : \text{TargetTerm } ([S]) \mapsto_{\parallel (\text{STCal Source Target})*} \ Q' \\
  \qquad \text{and } A2 : P' \leq_{\parallel RT} Q' \\
  \qquad \text{using } \text{bisimT} \\
  \qquad \text{by blast} \\
\end{array} \]

\[ \text{from refS } A2 \text{ have } P' \leq_{\parallel R} \text{SRel, TRel} > Q' \]

\[ \begin{array}{l}
  \text{by (simp add: STCal-steps(1))} \\
  \text{with } A1 \text{ show } \exists Q'. \text{ TargetTerm } ([S]) \mapsto_{\parallel (\text{STCal Source Target})*} \ Q' \land P' \leq_{\parallel R} \text{SRel, TRel} > Q' \\
  \qquad \text{by blast} \\
\end{array} \]

\[ \text{next} \]

\[ \text{case (source } S1 S2) \]

\[ \text{assume } \text{SourceTerm } S1 \mapsto_{\parallel (\text{STCal Source Target})*} \ P' \]

\[ \text{from this obtain } S1' \text{ where } B1 : S1' \in S \ P' \land B2 : S1 \mapsto_{\parallel \text{Source} \ S1' \]

\[ \begin{array}{l}
  \text{by (auto simp add: STCal-steps(1))} \\
  \text{assume } (S1, S2) \in SRel \\
  \text{with } B2 \text{ bisimS obtain } S2' \text{ where } B3 : S2 \mapsto_{\parallel \text{Source} \ S2' \land B4 : (S1', S2') \in SRel \\
  \qquad \text{by blast} \\
\end{array} \]

\[ \text{from } B3 \text{ have } \text{SourceTerm } S2 \mapsto_{\parallel (\text{STCal Source Target})* (\text{SourceTerm } S2') \\
\begin{array}{l}
  \text{by (simp add: STCal-steps(1))} \\
  \text{moreover from } B1 B4 \text{ have } P' \leq_{\parallel R} \text{SRel, TRel} > \text{SourceTerm } S2' \\
  \text{by (simp add: indRelRSTPO.source) \\
  \text{ultimately show } \exists Q'. \text{ SourceTerm } S2 \mapsto_{\parallel (\text{STCal Source Target})*} \ Q' \land P' \leq_{\parallel R} \text{SRel, TRel} > Q' \\
  \qquad \text{by blast} \\
\end{array} \]

\[ \text{next} \]

\[ \text{case (target } T1 T2) \]

\[ \text{assume } (T1, T2) \in TRel \\
  \text{hence } \text{TargetTerm } T1 \leq_{\parallel RT} TRel > \text{TargetTerm } T2 \\
  \text{by (rule indRelRSTPO.target) \\
  \text{moreover assume } \text{TargetTerm } T1 \mapsto_{\parallel (\text{STCal Source Target})*} \ P' \\
  \text{ultimately obtain } Q' \text{ where } C1 : \text{TargetTerm } T2 \mapsto_{\parallel (\text{STCal Source Target})*} \ Q' \\
  \qquad \text{and } C2 : P' \leq_{\parallel RT} Q' \\
  \qquad \text{using } \text{bisimT} \\
  \qquad \text{by blast} \\
\end{array} \]

\[ \text{from refS } C2 \text{ have } P' \leq_{\parallel R} \text{SRel, TRel} > Q' \\
\begin{array}{l}
  \text{by (simp add: indRelRSTPO.to-indRelRSTPO) \\
  \text{with } C1 \text{ show } \exists Q'. \text{ TargetTerm } T2 \mapsto_{\parallel (\text{STCal Source Target})*} \ Q' \land P' \leq_{\parallel R} \text{SRel, TRel} > Q' \\
  \qquad \text{by blast} \\
\end{array} \]

\[ \text{next} \]

\[ \text{case (trans } P Q R) \]

\[ \text{assume } P \mapsto_{\parallel (\text{STCal Source Target})*} P' \\
\begin{array}{l}
  \text{and } P'. P \mapsto_{\parallel (\text{STCal Source Target})*} P' \\
  \qquad \Rightarrow \exists Q'. Q \mapsto_{\parallel (\text{STCal Source Target})*} Q' \land P' \leq_{\parallel R} \text{SRel, TRel} > Q' \\
  \text{from this obtain } Q' \text{ where } D1 : Q \mapsto_{\parallel (\text{STCal Source Target})*} Q' \land D2 : P' \leq_{\parallel R} \text{SRel, TRel} > Q' \\
  \text{by blast} \\
  \text{assume } \land Q'. Q \mapsto_{\parallel (\text{STCal Source Target})*} Q' \\
  \qquad \Rightarrow \exists R'. R \mapsto_{\parallel (\text{STCal Source Target})*} R' \land Q' \leq_{\parallel R} \text{SRel, TRel} > R' \\
  \text{with } D1 \text{ obtain } R' \text{ where } D3 : R \mapsto_{\parallel (\text{STCal Source Target})*} R' \land D4 : Q' \leq_{\parallel R} \text{SRel, TRel} > R' \\
  \text{by blast} \\
  \text{from } D2 D4 \text{ have } P' \leq_{\parallel R} \text{SRel, TRel} > R' \\
  \text{by (rule indRelRSTPO.trans) \\
  \text{with } D3 \text{ show } \exists R'. R \mapsto_{\parallel (\text{STCal Source Target})*} R' \land P' \leq_{\parallel R} \text{SRel, TRel} > R' \\
  \qquad \text{by blast} \\
\end{array} \]

\text{qed} 

\text{next} 

149
fix \( P \), \( Q \), \( Q' \)

**assume** \( P \leq_{\{H\}R<\text{SRel}, \text{TRel}> Q \text{ and } Q \rightarrow (\text{STCal Source Target})* Q' \)

**thus** \( \exists P'. \ P \rightarrow (\text{STCal Source Target})* P' \wedge P' \leq_{\{H\}R<\text{SRel}, \text{TRel}> Q' \)

**proof** (induct arbitrary: \( Q' \))

**case** \((\text{encR} \ S)\)

**have** \( \exists \text{SourceTerm} \ S \leq_{\{H\}R<\text{TRel}> \text{TargetTerm} ([S]) \) 

by \((\text{rule indRelRTPO,encR})\)

moreover **assume** \( \exists \text{SourceTerm} \ ([S]) \rightarrow (\text{STCal Source Target})* Q' \)

ultimately **obtain** \( P' \) where \( E1: \exists \text{SourceTerm} \ S \rightarrow (\text{STCal Source Target})* P' \)

\( \text{and } E2: \ P' \leq_{\{H\}R<\text{TRel}> Q' \)

using \( \text{bisimT} \)

by \( \text{blast} \)

from \( \text{reflS} \ E2 \) **have** \( P' \leq_{\{H\}R<\text{SRel}, \text{TRel}> Q' \)

by \((\text{simp add: indRelRTPO-to-indRelRSTPO})\)

with \( E1 \) **show** \( \exists P'. \ SourceTerm \ S \rightarrow (\text{STCal Source Target})* P' \wedge P' \leq_{\{H\}R<\text{SRel}, \text{TRel}> Q' \)

by \( \text{blast} \)

next

**case** \( (\text{source} \ S1 \ S2) \)

**assume** \( \exists \text{SourceTerm} \ S2 \rightarrow (\text{STCal Source Target})* \) \( Q' \)

from \( \text{this} \) **obtain** \( S2' \) **where** \( F1: \ S2' \in S \ Q' \) **and** \( F2: \ S2 \rightarrow \) \( \text{Source} \) \( S2' \)

by \((\text{auto simp add: STCal-steps(1)})\)

**assume** \( \exists (S1, S2) \in \text{SRel} \)

with \( F2 \) **obtain** \( S1' \) **where** \( F3: \ S1 \rightarrow \) \( \text{Source} \) \( S1' \) **and** \( F4: \ (S1', S2') \in \text{SRel} \)

by \( \text{blast} \)

from \( F3 \) **have** \( \exists \text{SourceTerm} \ S1 \rightarrow (\text{STCal Source Target})* \ (\text{SourceTerm} \ S1') \)

by \((\text{simp add: STCal-steps(1)})\)

moreover from \( F1 \) \( F4 \) **have** \( \exists \text{SourceTerm} \ S1' \leq_{\{H\}R<\text{TRel}> Q' \)

by \((\text{simp add: indRelRSTPO,source})\)

ultimately **show** \( \exists P'. \ SourceTerm \ S1 \rightarrow (\text{STCal Source Target})* P' \wedge P' \leq_{\{H\}R<\text{TRel}> Q' \)

by \( \text{blast} \)

next

**case** \( (\text{target} \ T1 \ T2) \)

**assume** \( \exists (T1, T2) \in \text{TRel} \)

**hence** \( \exists \text{TargetTerm} \ T1 \leq_{\{H\}R<\text{TRel}> \text{TargetTerm} \ T2 \)

by \((\text{rule indRelRTPO,target})\)

moreover **assume** \( \exists \text{TargetTerm} \ T2 \rightarrow (\text{STCal Source Target})* \) \( Q' \)

ultimately **obtain** \( P' \) **where** \( G1: \ \exists \text{TargetTerm} \ T1 \rightarrow (\text{STCal Source Target})* P' \)

\( \text{and } G2: \ P' \leq_{\{H\}R<\text{TRel}> Q' \)

using \( \text{bisimT} \)

by \( \text{blast} \)

from \( \text{reflS} \ E2 \) **have** \( P' \leq_{\{H\}R<\text{SRel}, \text{TRel}> Q' \)

by \((\text{simp add: indRelRTPO-to-indRelRSTPO})\)

with \( G1 \) **show** \( \exists P'. \ TargetTerm \ T1 \rightarrow (\text{STCal Source Target})* P' \wedge P' \leq_{\{H\}R<\text{TRel}> Q' \)

by \( \text{blast} \)

next

**case** \( (\text{trans} \ P \ Q \ R \ R') \)

**assume** \( \forall P. \ R \rightarrow (\text{STCal Source Target})* R' \)

\( \text{and } \exists \forall P. \ R \rightarrow (\text{STCal Source Target})* R' \)

\( \Rightarrow \exists Q'. \ Q \rightarrow (\text{STCal Source Target})* Q' \wedge Q' \leq_{\{H\}R<\text{SRel}, \text{TRel}> R' \)

from \( \text{this} \) **obtain** \( Q' \) **where** \( H1: \ ) \( Q \rightarrow (\text{STCal Source Target})* Q' \) **and** \( H2: \ ) \( Q' \leq_{\{H\}R<\text{TRel}> R' \)

by \( \text{blast} \)

**assume** \( \exists \forall Q'. \ Q \rightarrow (\text{STCal Source Target})* Q' \)

\( \Rightarrow \exists P'. \ P \rightarrow (\text{STCal Source Target})* P' \wedge P' \leq_{\{H\}R<\text{SRel}, \text{TRel}> Q' \)

with \( H1 \) **obtain** \( P' \) **where** \( H3: \ P \rightarrow (\text{STCal Source Target})* P' \) **and** \( H4: \ P' \leq_{\{H\}R<\text{TRel}> Q' \)

by \( \text{blast} \)

from \( H4 \) \( H2 \) **have** \( P' \leq_{\{H\}R<\text{SRel}, \text{TRel}> R' \)

by \((\text{rule indRelRSTPO,trans})\)

with \( H3 \) **show** \( \exists P'. \ P \rightarrow (\text{STCal Source Target})* P' \wedge P' \leq_{\{H\}R<\text{TRel}> R' \)

by \( \text{blast} \)

\( \text{qed} \)

\( \text{qed} \)
6 Success Sensitiveness and Barbs

To compare the abstract behavior of two terms, often some notion of success or successful termination is used. Daniele Gorla assumes a constant process (similar to the empty process) that represents successful termination in order to compare the behavior of source terms with their literal translations. Then an encoding is success sensitive if, for all source terms $S$, $S$ reaches success iff the translation of $S$ reaches success. Successful termination can be considered as some special kind of barb. Accordingly we generalize successful termination to the respectation of an arbitrary subset of barbs. An encoding respects a set of barbs if, for every source term $S$ and all considered barbs $a$, $S$ reaches $a$ iff the translation of $S$ reaches $a$.

Abbreviation (in encoding-wrt-barbs) enc-weakly-preserves-barb-set :: 'barbs \Rightarrow bool where
enc-weakly-preserves-barb-set Barbs \equiv enc-preserves-binary-pred (\lambda P \ a. a \in Barbs \land P \downarrow a)

Abbreviation (in encoding-wrt-barbs) enc-weakly-preserves-barbs :: bool where
enc-weakly-preserves-barbs \equiv enc-preserves-binary-pred (\lambda P a. P \downarrow a)

Lemma (in encoding-wrt-barbs) enc-weakly-preserves-barbs-and-barb-set:
shows enc-weakly-preserves-barbs = (\forall Barbs. enc-weakly-preserves-barb-set Barbs)
by blast

Abbreviation (in encoding-wrt-barbs) enc-weakly-reflects-barb-set :: 'barbs \Rightarrow bool where
enc-weakly-reflects-barb-set Barbs \equiv enc-reflects-binary-pred (\lambda P a. a \in Barbs \land P \downarrow a)

Abbreviation (in encoding-wrt-barbs) enc-weakly-reflects-barbs :: bool where
enc-weakly-reflects-barbs \equiv enc-reflects-binary-pred (\lambda P a. P \downarrow a)

Lemma (in encoding-wrt-barbs) enc-weakly-reflects-barbs-and-barb-set:
shows enc-weakly-reflects-barbs = (\forall Barbs. enc-weakly-reflects-barb-set Barbs)
by blast

Abbreviation (in encoding-wrt-barbs) enc-weakly-respects-barb-set :: 'barbs \Rightarrow bool where

Abbreviation (in encoding-wrt-barbs) enc-weakly-respects-barbs :: bool where
enc-weakly-respects-barbs \equiv enc-weakly-preserves-barbs \land enc-weakly-reflects-barbs

Lemma (in encoding-wrt-barbs) enc-weakly-respects-barbs-and-barb-set:
shows enc-weakly-respects-barbs = (\forall Barbs. enc-weakly-respects-barb-set Barbs)

Proof

- have (\forall Barbs. enc-weakly-respects-barb-set Barbs)
  = (\forall Barbs. (\forall S x. x \in Barbs \land S \downarrow SWB > x \longrightarrow [S] \downarrow TWB > x)
  \land (\forall S x. x \in Barbs \land [S] \downarrow TWB > x \longrightarrow S \downarrow SWB > x))
  by simp

- hence (\forall Barbs. enc-weakly-respects-barb-set Barbs)
  = ((\forall Barbs. enc-weakly-preserves-barb-set Barbs)
  \land (\forall Barbs. enc-weakly-reflects-barb-set Barbs))

- apply simp by fast

Thus \thesis

- apply simp by blast

Qed

An encoding strongly respects some set of barbs if, for every source term $S$ and all considered barbs
a, S has a iff the translation of S has a.

**abbreviation** (in encoding-wrt-barbs) enc-preserves-barb-set :: 'barbs set ⇒ bool where
cenc-preserves-barb-set Barbs ≡ enc-preserves-binary-pred (λP a. a ∈ Barbs ∧ P ↓ a)

**abbreviation** (in encoding-wrt-barbs) enc-preserves-barbs :: bool where
cenc-preserves-barbs ≡ enc-preserves-binary-pred (λP a. P ↓ a)

**lemma** (in encoding-wrt-barbs) enc-preserves-barbs-and-barb-set:
shows enc-preserves-barbs = (∀ Barbs. enc-preserves-barb-set Barbs)
by blast

**abbreviation** (in encoding-wrt-barbs) enc-reflects-barb-set :: 'barbs set ⇒ bool where
cenc-reflects-barb-set Barbs ≡ enc-reflects-binary-pred (λP a. a ∈ Barbs ∧ P ↓ a)

**abbreviation** (in encoding-wrt-barbs) enc-reflects-barbs :: bool where
cenc-reflects-barbs ≡ enc-reflects-binary-pred (λP a. P ↓ a)

**lemma** (in encoding-wrt-barbs) enc-reflects-barbs-and-barb-set:
shows enc-reflects-barbs = (∀ Barbs. enc-reflects-barb-set Barbs)
by blast

**abbreviation** (in encoding-wrt-barbs) enc-respects-barb-set :: 'barbs set ⇒ bool where
cenc-respects-barb-set Barbs ≡ enc-preserves-barb-set Barbs ∧ enc-reflects-barb-set Barbs

**abbreviation** (in encoding-wrt-barbs) enc-respects-barbs :: bool where
cenc-respects-barbs ≡ enc-preserves-barbs ∧ enc-reflects-barbs

**lemma** (in encoding-wrt-barbs) enc-respects-barbs-and-barb-set:
shows enc-respects-barbs = (∀ Barbs. enc-respects-barb-set Barbs)
proof −
  have (∀ Barbs. enc-respects-barb-set Barbs)
  = ((∀ Barbs. enc-preserves-barb-set Barbs) ∧ (∀ Barbs. enc-reflects-barb-set Barbs))
  apply simp by fast
  thus ?thesis
  apply simp by blast
qed

An encoding (weakly) preserves barbs iff (1) there exists a relation, like indRelR, that relates source
terms and their literal translations and preserves (reachability/existence of barbs, or (2) there exists
a relation, like indRelL, that relates literal translations and their source terms and reflects (reachability/
existence of barbs.

**lemma** (in encoding-wrt-barbs) enc-weakly-preserves-barb-set-iff-source-target-rel:
fixes Barbs :: 'barbs set
and TRel :: ('procT × 'procT) set
shows enc-weakly-preserves-barb-set Barbs
= (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ rel-weakly-preserves-barb-set Rel (STCalWB SWB TWB) Barbs)
using enc-preserves-binary-pred-iff-source-target-rel-preserves-binary-pred[where
Pred=λP a. a ∈ Barbs ∧ P ↓ <STCalWB SWB TWB> a] STCalWB-reachesBarbST
by simp

**lemma** (in encoding-wrt-barbs) enc-weakly-preserves-barbs-iff-source-target-rel:
shows enc-weakly-preserves-barbs
= (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ rel-weakly-preserves-barbs Rel (STCalWB SWB TWB))
using enc-preserves-binary-pred-iff-source-target-rel-preserves-binary-pred[where
Pred=λP a. P ↓ <STCalWB SWB TWB> a] STCalWB-reachesBarbST
by simp
lemma (in encoding-wrt-barbs) enc-preserves-barb-set-iff-source-target-rel:
  fixes Barbs :: 'barbs set
  shows enc-preserves-barb-set Barbs
    = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
      ∧ rel-preserves-barb-set Rel (STCalWB SWB TWB) Barbs)
  using enc-preserves-binary-pred-iff-source-target-rel-preserves-binary-pred[where
    Pred=\(\lambda P. a. a \in Barbs \land P\downarrow<STCalWB SWB TWB>a\) STCalWB-hasBarbST]
    by simp

lemma (in encoding-wrt-barbs) enc-preserves-barbs-iff-source-target-rel:
  shows enc-preserves-barbs
    = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
      ∧ rel-preserves-barbs Rel (STCalWB SWB TWB))
  using enc-preserves-binary-pred-iff-source-target-rel-preserves-binary-pred[where
    Pred=\(\lambda P. a. P\downarrow<STCalWB SWB TWB>a\) STCalWB-hasBarbST]
    by simp

lemma (in encoding-wrt-barbs) enc-weakly-reflects-barb-set-iff-source-target-rel:
  fixes Barbs :: 'barbs set
  shows enc-weakly-reflects-barb-set Barbs
    = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
      ∧ rel-weakly-reflects-barb-set Rel (STCalWB SWB TWB) Barbs)
  using enc-reflects-binary-pred-iff-source-target-rel-reflects-binary-pred[where
    Pred=\(\lambda P. a. P\downarrow<STCalWB SWB TWB>a\) STCalWB-reachesBarbST]
    by simp

lemma (in encoding-wrt-barbs) enc-reflects-barb-set-iff-source-target-rel:
  shows enc-reflects-barb-set
    = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
      ∧ rel-reflects-barb-set Rel (STCalWB SWB TWB) Barbs)
  using enc-reflects-binary-pred-iff-source-target-rel-reflects-binary-pred[where
    Pred=\(\lambda P. a. P\downarrow<STCalWB SWB TWB>a\) STCalWB-hasBarbST]
    by simp

lemma (in encoding-wrt-barbs) enc-reflects-barbs-iff-source-target-rel:
  shows enc-reflects-barbs
    = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
      ∧ rel-reflects-barbs Rel (STCalWB SWB TWB))
  using enc-reflects-binary-pred-iff-source-target-rel-reflects-binary-pred[where
    Pred=\(\lambda P. a. P\downarrow<STCalWB SWB TWB>a\) STCalWB-reachesBarbST]
    by simp

lemma (in encoding-wrt-barbs) enc-preserves-barbs-iff-source-target-rel:
  shows enc-preserves-barbs
    = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
      ∧ rel-preserves-barbs Rel (STCalWB SWB TWB) Barbs)
  using enc-preserves-binary-pred-iff-source-target-rel-preserves-binary-pred[where
    Pred=\(\lambda P. a. a \in Barbs \land P\downarrow<STCalWB SWB TWB>a\) STCalWB-hasBarbST]
    by simp

lemma (in encoding-wrt-barbs) enc-reflects-barbs-iff-source-target-rel:
  shows enc-reflects-barbs
    = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
      ∧ rel-reflects-barbs Rel (STCalWB SWB TWB))
  using enc-reflects-binary-pred-iff-source-target-rel-reflects-binary-pred[where
    Pred=\(\lambda P. a. P\downarrow<STCalWB SWB TWB>a\) STCalWB-hasBarbST]
    by simp

An encoding (weakly) reflects barbs iff (1) there exists a relation, like indRelR, that relates source terms and their literal translations and reflects (reachability/.existence of barbs, or (2) there exists a relation, like indRelL, that relates literal translations and their source terms and preserves (reachability/.existence of barbs, or (3) there exists a relation, like indRel, that relates source terms and their literal translations in both directions and respects (reachability/.existence of barbs.
lemma (in encoding-wrt-barbs) enc-weakly-respects-barb-set-iff-source-target-rel:

fixes Barbs :: 'barbs set

shows enc-weakly-respects-barb-set Barbs
= (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
∧ rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) Barbs)

using enc-respects-binary-pred-iff-source-target-rel-respects-binary-pred-encR[where
Pred=λP a. a ∈ Barbs ∧ P↓<STCalWB SWB TWB>a] STCalWB-reachesBarbST

by simp

lemma (in encoding-wrt-barbs) enc-weakly-respects-barb-set-iff-source-target-rel:

shows enc-weakly-respects-barbs
= (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
∧ rel-weakly-respects-barbs Rel (STCalWB SWB TWB))

using enc-respects-binary-pred-iff-source-target-rel-respects-binary-pred-encR[where
Pred=λP a. P↓<STCalWB SWB TWB>a] STCalWB-reachesBarbST

by simp

lemma (in encoding-wrt-barbs) enc-respects-barb-set-iff-source-target-rel:

shows enc-respects-barbs
= (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
∧ rel-respects-barbs Rel (STCalWB SWB TWB))

using enc-respects-binary-pred-iff-source-target-rel-respects-binary-pred-encR[where
Pred=λP a. P↓<STCalWB SWB TWB>a] STCalWB-hasBarbST

by simp

Accordingly an encoding is success sensitive iff there exists such a relation between source and target terms that weakly respects the barb success.

lemma (in encoding-wrt-barbs) success-sensitive-cond:

fixes success :: 'barbs

shows enc-weakly-respects-barb-set {success} = (∀ S. S↓<SWB>success ⟷ [S]↓<TWB>success)

by auto

lemma (in encoding-wrt-barbs) success-sensitive-iff-source-target-rel-weakly-respects-success:

fixes success :: 'barbs

shows enc-weakly-respects-barb-set {success}
= (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
∧ rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) {success})

by (rule enc-weakly-respects-barb-set-iff-source-target-rel[where Barbs={success}])+

lemma (in encoding-wrt-barbs) success-sensitive-iff-source-target-rel-respects-success:

fixes success :: 'barbs

shows enc-respects-barb-set {success}
= (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
∧ rel-respects-barb-set Rel (STCalWB SWB TWB) {success})

by (rule enc-respects-barb-set-iff-source-target-rel[where Barbs={success}])

end

theory DivergenceReflection

imports SourceTargetRelation

begin
Divergence Reflection

Divergence reflection forbids for encodings that introduce loops of internal actions. Thus they determine the practicability of encodings in particular with respect to implementations. An encoding reflects divergence if each loop in a target term result from the translation of a divergent source term.

abbreviation (in encoding) enc-preserves-divergence :: bool where
enc-preserves-divergence ≡ enc-preserves-pred (λP. P ↦→ STω)

lemma (in encoding) divergence-preservation-cond:
  shows enc-preserves-divergence = (∀ S. S ↦→ (Source)ω → [S] ↦→ (Target)ω)
  by simp

abbreviation (in encoding) enc-reflects-divergence :: bool where
enc-reflects-divergence ≡ enc-reflects-pred (λP. P ↦→ STω)

lemma (in encoding) divergence-reflection-cond:
  shows enc-reflects-divergence = (∀ S. [S] ↦→ (Target)ω ←→ S ↦→ (Source)ω)
  by simp

Apart from divergence reflection we consider divergence respection. An encoding respects divergence if each divergent source term is translated into a divergent target term and each divergent target term result from the translation of a divergent source term.

abbreviation (in encoding) enc-respects-divergence :: bool where
enc-respects-divergence ≡ enc-respects-pred (λP. P ↦→ STω)

lemma (in encoding) divergence-respection-cond:
  shows enc-respects-divergence = (∀ S. TargetTerm ([S]) ↦→ S ↦→ (Source)ω)
  by auto

abbreviation rel-preserves-divergence :: ('proc × 'proc) set ⇒ 'proc processCalculus ⇒ bool
  where
rel-preserves-divergence Rel Cal ≡ rel-preserves-pred Rel (λP. P ↦→ (Cal)ω)

abbreviation rel-reflects-divergence :: ('proc × 'proc) set ⇒ 'proc processCalculus ⇒ bool
  where
rel-reflects-divergence Rel Cal ≡ rel-reflects-pred Rel (λP. P ↦→ (Cal)ω)

An encoding preserves divergence iff (1) there exists a relation that relates source terms and their literal translations and preserves divergence, or (2) there exists a relation that relates literal translations and their source terms and reflects divergence.

lemma (in encoding) divergence-preservation-iff-source-target-rel-preserves-divergence:
  shows enc-preserves-divergence
  = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel ∧ rel-preserves-divergence Rel (STCal Source Target)))
  using enc-preserves-pred-iff-source-target-rel-preserves-pred(t)[where Pred=λP. P ↦→ STω]
  by simp

lemma (in encoding) divergence-preservation-iff-source-target-rel-reflects-divergence:
  shows enc-preserves-divergence
  = (∃ Rel. (∀ S. (TargetTerm ([S]), SourceTerm S) ∈ Rel ∧ rel-reflects-divergence Rel (STCal Source Target)))
An encoding reflects divergence iff (1) there exists a relation that relates source terms and their literal translations and reflects divergence, or (2) there exists a relation that relates literal translations and their source terms and preserves divergence.

**Lemma (in encoding) divergence-reflection-iff-source-target-rel-reflects-divergence:**

**shows**\( enc\text{-}reflects\text{-}divergence = (\exists \text{Rel}. (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}) \land \text{rel\text{-}reflects\text{-}divergence Rel } (\text{STCal Source Target})) \)

**using**\( enc\text{-}reflects\text{-}pred\text{-}iff\text{-}source\text{-}target\text{-}rel\text{-}reflects\text{-}pred \)

by simp

An encoding respects divergence iff there exists a relation that relates source terms and their literal translations in both directions and respects divergence.

**Lemma (in encoding) divergence-respection-iff-source-target-rel-respects-divergence:**

**shows**\( enc\text{-}respects\text{-}divergence = (\exists \text{Rel}. (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}) \land \text{rel\text{-}respects\text{-}divergence Rel } (\text{STCal Source Target})) \)

**and**\( enc\text{-}respects\text{-}divergence = (\exists \text{Rel}. (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \land (\text{TargetTerm } ([S]), \text{SourceTerm } S) \in \text{Rel}) \land \text{rel\text{-}respects\text{-}divergence Rel } (\text{STCal Source Target})) \)

**proof**

**show**\( enc\text{-}respects\text{-}divergence = (\exists \text{Rel}. (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}) \land \text{rel\text{-}respects\text{-}divergence Rel } (\text{STCal Source Target})) \)

**using**\( enc\text{-}respects\text{-}pred\text{-}iff\text{-}source\text{-}target\text{-}rel\text{-}respects\text{-}pred\text{-}encR \)

by simp

next

**show**\( enc\text{-}respects\text{-}divergence = (\exists \text{Rel}. (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \land (\text{TargetTerm } ([S]), \text{SourceTerm } S) \in \text{Rel}) \land \text{rel\text{-}respects\text{-}divergence Rel } (\text{STCal Source Target})) \)

**using**\( enc\text{-}respects\text{-}pred\text{-}iff\text{-}source\text{-}target\text{-}rel\text{-}respects\text{-}pred\text{-}encRL \)

by simp

qed

end

theory OperationalCorrespondence

imports SourceTargetRelation

begin

8 Operational Correspondence

We consider different variants of operational correspondence. This criterion consists of a completeness and a soundness condition and is often defined with respect to a relation TRel on target terms. Operational completeness modulo TRel ensures that an encoding preserves source term behaviour modulo TRel by requiring that each sequence of source term steps can be mimicked by its translation such that the respective derivatives are related by TRel.
abbreviation (in encoding) operational-complete :: (‘procT × ‘procT) set ⇒ bool where
operational-complete TRel ≡ ∀ S S’. S ↦−→Source* S’ → (∃ T. [S] ↦−→Target* T ∧ ([S’], T) ∈ TRel)

We call an encoding strongly operational complete modulo TRel if each source term step has to be mimicked by single target term step of its translation.

abbreviation (in encoding) strongly-operational-complete :: (‘procT × ‘procT) set ⇒ bool where
strongly-operational-complete TRel ≡ ∀ S S’. S ↦−→Source* S’ → (∃ T. [S] ↦−→Target* T ∧ ([S’], T) ∈ TRel)

Operational soundness ensures that the encoding does not introduce new behaviour. An encoding is weakly operational sound modulo TRel if each sequence of target term steps is part of the translation of a sequence of source term steps such that the derivatives are related by TRel. It allows for intermediate states on the translation of source term step that are not the result of translating a source term.

abbreviation (in encoding) weakly-operational-sound :: (‘procT × ‘procT) set ⇒ bool where
weakly-operational-sound TRel ≡ ∀ S T. [S] ↦−→Target* T → (∃ S’ T’. S ↦−→Source* S’ ∧ T ↦−→Target* T’ ∧ ([S’], T’) ∈ TRel)

And encoding is operational sound modulo TRel if each sequence of target term steps is the translation of a sequence of source term steps such that the derivatives are related by TRel. This criterion does not allow for intermediate states, i.e., does not allow to reach target term from an encoded source term that is not related by TRel to the translation of a source term.

abbreviation (in encoding) operational-sound :: (‘procT × ‘procT) set ⇒ bool where
operational-sound TRel ≡ ∀ S T. [S] ↦−→Target* T → (∃ S’ T’. S ↦−→Source* S’ ∧ ([S’], T) ∈ TRel)

Strong operational soundness modulo TRel is a stricter variant of operational soundness, where a single target term step has to be mapped on a single source term step.

abbreviation (in encoding) strongly-operational-sound :: (‘procT × ‘procT) set ⇒ bool where
strongly-operational-sound TRel ≡ ∀ S T. [S] ↦−→Target* T → (∃ S’ T’. S ↦−→Source* S’ ∧ ([S’], T) ∈ TRel)

An encoding is weakly operational corresponding modulo TRel if it is operational complete and weakly operational sound modulo TRel.

abbreviation (in encoding) weakly-operational-corresponding :: (‘procT × ‘procT) set ⇒ bool where
weakly-operational-corresponding TRel ≡ operational-complete TRel ∧ weakly-operational-sound TRel

Operational correspondence modulo is the combination of operational completeness and operational soundness modulo TRel.

abbreviation (in encoding) operational-corresponding :: (‘procT × ‘procT) set ⇒ bool where
operational-corresponding TRel ≡ operational-complete TRel ∧ operational-sound TRel

An encoding is strongly operational corresponding modulo TRel if it is strongly operational complete and strongly operational sound modulo TRel.

abbreviation (in encoding) strongly-operational-corresponding :: (‘procT × ‘procT) set ⇒ bool where
strongly-operational-corresponding TRel ≡ strongly-operational-complete TRel ∧ strongly-operational-sound TRel

8.1 Trivial Operational Correspondence Results

Every encoding is (weakly) operational corresponding modulo the all relation on target terms.

lemma (in encoding) operational-correspondence-modulo-all-relation:
  shows operational-complete {(T1, T2). True}
and weakly-operational-sound \{ (T1, T2). \text{True} \}
and operational-sound \{ (T1, T2). \text{True} \}
using steps-refl[where \text{Cal} = \text{Source}] steps-refl[where \text{Cal} = \text{Target}]
by blast+

\text{lemma all-relation-is-weak-reduction-bisimulation:}
\text{fixes Cal :: 'a processCalculus}
\text{shows weak-reduction-bisimulation \{ (a, b). \text{True} \} Cal}
using steps-refl[where \text{Cal} = \text{Cal}]
by blast

\text{lemma (in encoding) operational-correspondence-modulo-some-target-relation:}
\text{shows } \exists \text{TRel. weakly-operational-corresponding TRel}
and \exists \text{TRel. operational-corresponding TRel}
and \exists \text{TRel. weakly-operational-corresponding TRel} \land \text{weak-reduction-bisimulation TRel Target}
and \exists \text{TRel. operational-corresponding TRel} \land \text{weak-reduction-bisimulation TRel Target}
using operational-correspondence-modulo-all-relation
all-relation-is-weak-reduction-bisimulation[where \text{Cal} = \text{Target}]
by blast+

Strong operational correspondence requires that source can perform a step iff their translations can perform a step.

\text{lemma (in encoding) strong-operational-correspondence-modulo-some-target-relation:}
\text{shows } (\exists \text{TRel. strongly-operational-corresponding TRel})
\text{and } (\exists \text{TRel. operational-corresponding TRel})
\text{and } \exists \text{TRel. weakly-operational-corresponding TRel} \land \text{weak-reduction-bisimulation TRel Target}
\text{and } \exists \text{TRel. operational-corresponding TRel} \land \text{weak-reduction-bisimulation TRel Target}
using all-relation-is-weak-reduction-bisimulation[where \text{Cal} = \text{Target}]
by blast

\text{proof --}
\text{have A1: } \exists \text{TRel. strongly-operational-corresponding TRel}
\implies \forall S. (\exists S'. S \mapsto \text{Source } S') \iff (\exists T. [S] \mapsto \text{Target } T)
by blast

\text{moreover have A2: } \forall S. (\exists S'. S \mapsto \text{Source } S') \iff (\exists T. [S] \mapsto \text{Target } T)
\implies \exists \text{TRel. strongly-operational-corresponding TRel}
\land \text{weak-reduction-bisimulation TRel Target}
\text{proof --}
\text{assume } \forall S. (\exists S'. S \mapsto \text{Source } S') \iff (\exists T. [S] \mapsto \text{Target } T)
\text{hence strongly-operational-corresponding \{ (T1, T2). \text{True} \}}
by simp

\text{thus } \exists \text{TRel. strongly-operational-corresponding TRel}
\land \text{weak-reduction-bisimulation TRel Target}
using all-relation-is-weak-reduction-bisimulation[where \text{Cal} = \text{Target}]
by blast
qed

\text{ultimately show } (\exists \text{TRel. strongly-operational-corresponding TRel}
\land \text{weak-reduction-bisimulation TRel Target})
\implies (\forall S. (\exists S'. S \mapsto \text{Source } S') \iff (\exists T. [S] \mapsto \text{Target } T))
by blast

from A1 A2 show \( \exists \text{TRel. strongly-operational-corresponding TRel} 
\implies (\forall S. (\exists S'. S \mapsto \text{Source } S') \iff (\exists T. [S] \mapsto \text{Target } T)) 
by blast
qed

\textbf{8.2 (Strong) Operational Completeness vs (Strong) Simulation}

An encoding is operational complete modulo a weak simulation on target terms TRel iff there is a relation, like indRelRTPO, that relates at least all source terms to their literal translations, includes TRel, and is a weak simulation.

\textbf{lemma (in encoding) weak-reduction-simulation-impl-OCom:}
proof \( \exists S' \) 
from \( A1 \) have \((\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel} \)
by \simp
moreover assume \( S \rightarrow(Source* S') \)

hence \( \text{SourceTerm} S \rightarrow(STCal \text{ Source Target})* (\text{SourceTerm} S') \)
by \( \simp \) \( \text{simp add: STCal-steps(1)} \)

ultimately obtain \( Q' \) where \( A5: \text{TargetTerm} ([S]) \rightarrow(STCal \text{ Source Target})* Q' \)
and \( A6: (\text{SourceTerm} S', Q') \in \text{Rel} \)

using \( A3 \)
by \( \text{blast} \)
from \( A5 \) obtain \( T \) where \( A7: T \in T Q' \) and \( A8: [S] \rightarrow Target* T \)
by \( \text{(auto simp add: STCal-steps(2))} \)
from \( A2 \) \( A6 \) \( A7 \) have \(( [S'], T) \in \text{TRel}^* \)
by \( \simp \)
with \( A8 \) show \( \exists T. [S] \rightarrow Target* T \land ([S'], T) \in \text{TRel}^* \)
by \( \text{blast} \)

qed

lemma (in encoding) \( \text{OCom-iff-indRelRTPO-is-weak-reduction-simulation} \):

fixes \( \text{TRel} :: (\text{'procT} \times \text{'procT}) \text{ set} \)

shows \( \text{(operational-complete (TRel*) \wedge weak-reduction-simulation (TRel^*) Target)} \)

proof \( \text{(rule iffI, erule conjE)} \)
assume oc \, \text{operational-complete (TRel*)}
and sim: \text{weak-reduction-simulation (TRel^*) Target}
show \( \text{weak-reduction-simulation (indRelRTPO TRel) (STCal Source Target)} \)

proof \( \text{clarify} \)
fix \( P Q P' \)
assume \( P \leq [\text{RT<TRel>} Q \land P \rightarrow(STCal \text{ Source Target})* P' \)
thus \( \exists Q', Q' \rightarrow(STCal \text{ Source Target})* Q' \land P' \leq [\text{RT<TRel>} Q' \)
proof \( \text{(induct arbitrary: P'} \)

case \( \text{encR} S \)
assume \( \text{SourceTerm} S \rightarrow(STCal \text{ Source Target})* P' \)
from \( \text{this} \) obtain \( S' \) where \( A1: S' \in S P' \) and \( A2: S \rightarrow Source* S' \)
by \( \text{(auto simp add: STCal-steps(1))} \)
from \( \text{oc A2} \) obtain \( T \) where \( A3: [S] \rightarrow Target* T \) and \( A4: ([S'], T) \in \text{TRel}^* \)
by \( \text{blast} \)
from \( A3 \) have \( \text{TargetTerm} ([S]) \rightarrow(STCal \text{ Source Target})* (\text{TargetTerm} T) \)
by \( \text{(simp add: STCal-steps(2))} \)
moreover have \( P' \leq [\text{RT<TRel>} TargetTerm T \)
proof \( \text{=} \)
from \( A4 \) have \( [S'] = T \lor ([S'], T) \in \text{TRel}^* \)
using \rtrancl-eq-or-rtrancl[of \([S'] T T \text{TRel} \]
by \( \text{blast} \)
moreover from \( A1 \) have \( A5: P' \leq [\text{RT<TRel>} TargetTerm ([S'] \)
by \( \text{(simp add: indRelRTPO.\text{encR})} \)
hence \( [S'] = T \Rightarrow P' \leq [\text{RT<TRel>} TargetTerm T \)
by \( \text{simp} \)
moreover have \( ([S'], T) \in \text{TRel}^* \Rightarrow P' \leq [\text{RT<TRel>} TargetTerm T \)
proof \( \text{induct} \)
assume \( ([S'], T) \in \text{TRel}^* \)
hence \( \text{TargetTerm} ([S']) \leq [\text{RT<TRel>} TargetTerm T \)
proof \( \text{induct} \)
fix \( T \)
assume \((\{S\}, T) \in TRel\)
thus \( \text{TargetTerm}(\{S\}) \preceq [\![RT\!]<TRel>] \text{TargetTerm} T \)
  by (rule indRelRTPO.target)

next
case \((\text{step} \ TQ \ TR)\)
assume \( \text{TargetTerm}(\{S\}) \preceq [\![RT\!]<TRel>] \text{TargetTerm} TQ \)
moreover assume \( (TQ, TR) \in TRel \)
hence \( \text{TargetTerm} TQ \preceq [\![RT\!]<TRel>] \text{TargetTerm} TR \)
  by (rule indRelRTPO.target)
ultimately show \( \text{TargetTerm}(\{S\}) \preceq [\![RT\!]<TRel>] \text{TargetTerm} TR \)
  by (rule indRelRTPO.trans)
qed
with \( A5 \) show \( P' \preceq [\![RT\!]<TRel>] \text{TargetTerm} T \)
  by (rule indRelRTPO.trans)
qed
ultimately
show \( \exists Q'. \text{TargetTerm}(\{S\}) \mapsto (\text{STCal Source Target})^* Q' \wedge P' \preceq [\![RT\!]<TRel>] Q' \)
  by blast

next
case \((\text{source} S)\)
then obtain \( S' \) where \( Bk: S' \in S \ P' \)
  by (auto simp add: STCal-steps(1))
hence \( P' \preceq [\![RT\!]<TRel>] P' \)
  by (simp add: indRelRTPO.source)
with source show \( \exists Q'. \text{SourceTerm} S \mapsto (\text{STCal Source Target})^* Q' \wedge P' \preceq [\![RT\!]<TRel>] Q' \)
  by blast

next
case \((\text{target} T1 \ T2)\)
assume \( \text{TargetTerm} T1 \mapsto (\text{STCal Source Target})^* P' \)
from this obtain \( T1' \) where \( C1: T1' \in T \ P' \) and \( C2: T1 \mapsto \text{Target}^* T1' \)
  by (auto simp add: STCal-steps(2))
assume \( (T1, T2) \in TRel \)
hence \( (T1, T2) \in TRel^+ \)
  by simp
with \( C2 \) sim obtain \( T2' \) where \( C3: T2 \mapsto \text{Target}^* T2' \)
  and \( C4: (T1', T2') \in TRel^+ \)
  by blast
from \( C3 \) have \( \text{TargetTerm} T2 \mapsto (\text{STCal Source Target})^* (\text{TargetTerm} T2') \)
  by (simp add: STCal-steps(2))
moreover from \( C4 \) have \( \text{TargetTerm} T1' \preceq [\![RT\!]<TRel>] \text{TargetTerm} T2' \)
proof induct
fix \( T2' \)
assume \( (T1', T2') \in TRel \)
thus \( \text{TargetTerm} T1' \preceq [\![RT\!]<TRel>] \text{TargetTerm} T2' \)
  by (rule indRelRTPO.target)
next
case \((\text{step} \ TQ \ TR)\)
assume \( \text{TargetTerm} T1' \preceq [\![RT\!]<TRel>] \text{TargetTerm} TQ \)
moreover assume \( (TQ, TR) \in TRel \)
hence \( \text{TargetTerm} TQ \preceq [\![RT\!]<TRel>] \text{TargetTerm} TR \)
  by (rule indRelRTPO.target)
ultimately show \( \text{TargetTerm} T1' \preceq [\![RT\!]<TRel>] \text{TargetTerm} TR \)
  by (rule indRelRTPO.trans)
qed
with \( C1 \) have \( P' \preceq [\![RT\!]<TRel>] \text{TargetTerm} T2' \)
  by simp
ultimately show \( \exists Q'. \text{TargetTerm} T2 \mapsto (\text{STCal Source Target})^* Q' \wedge P' \preceq [\![RT\!]<TRel>] Q' \)
by blast
next
case (trans $P$ $Q$ $R$)
assume $P$ $\rightsquigarrow$ ($\text{STCal Source Target}$) $P'$
and $\forall P'. P$ $\rightsquigarrow$ ($\text{STCal Source Target}$) $P'$
\[ \implies \exists Q'. Q$ $\rightsquigarrow$ ($\text{STCal Source Target}$) $Q' \land P' \lesssim_{\left[\left[\right]\right]}RT < TRel > Q' \]
from this obtain $Q'$ where $D_1$: $Q$ $\rightsquigarrow$ ($\text{STCal Source Target}$) $Q'$
and $D_2$: $P' \lesssim_{\left[\left[\right]\right]}RT < TRel > Q'$
by blast
assume $\forall Q'. Q$ $\rightsquigarrow$ ($\text{STCal Source Target}$) $Q'$
\[ \implies \exists R'. R$ $\rightsquigarrow$ ($\text{STCal Source Target}$) $R' \land Q' \lesssim_{\left[\left[\right]\right]}RT < TRel > R' \]
with $D_1$ obtain $R'$ where $D_3$: $R$ $\rightsquigarrow$ ($\text{STCal Source Target}$) $R'$
and $D_4$: $Q' \lesssim_{\left[\left[\right]\right]}RT < TRel > R'$
by blast
qed
qed
next
have $\forall S. \text{SourceTerm} S \lesssim_{\left[\left[\right]\right]}RT < TRel > \text{TargetTerm} (\left[\left[\right]\right])$
by (simp add: indRelRTPO_encR)
moreover have $\forall S T. \text{SourceTerm} S \lesssim_{\left[\left[\right]\right]}RT < TRel > \text{TargetTerm} T \rightsquigarrow (\left[\left[\right]\right], T) \in TRel^*$
using indRelRTPO-to-TRel(2)[where $TRel = TRel$] trans-closure-of-TRel-refl-cond
by simp
moreover assume sim: weak-reduction-simulation (indRelRTPO $TRel$) (STCal Source Target)
ultimately have operational-complete ($TRel^*$)
using weak-reduction-simulation-impl-OCom[where $Rel = indRelRTPO$ $TRel$ and $TRel = TRel$]
by simp
moreover from sim have weak-reduction-simulation ($TRel^*$) Target
using indRelRTPO-impl-TRel-is-weak-reduction-simulation[where $TRel = TRel$]
by simp
ultimately show operational-complete ($TRel^*$)
\[ \land \text{weak-reduction-simulation } Rel (\text{STCal Source Target}) \]
by simp
qed

lemma (in encoding) OCom-iff-weak-reduction-simulation:
fixes $TRel :: (\text{procT} 	imes \text{procT})$ set
shows (operational-complete ($TRel^*$))
\[ \land \text{weak-reduction-simulation } (\text{Rel}) (\text{TRel}) \]
proof (rule ifI, erule conjE)
have $\forall S. (\text{SourceTerm} S, \text{TargetTerm} (\left[\left[\right]\right])) \in indRelRTPO TRel$
by (simp add: indRelRTPO_encR)
moreover have $\forall T_1 T_2. (T_1, T_2) \in TRel \rightsquigarrow \text{TargetTerm} T_1 \lesssim_{\left[\left[\right]\right]}RT < TRel > \text{TargetTerm T2}$
by (simp add: indRelRTPO_target)
moreover have $\forall T_1 T_2. \text{TargetTerm} T_1 \lesssim_{\left[\left[\right]\right]}RT < TRel > \text{TargetTerm T2} \rightsquigarrow (T_1, T_2) \in TRel^*$
using indRelRTPO-to-TRel(4)[where $TRel = TRel$]
by simp
moreover have $\forall S T. \text{SourceTerm} S \lesssim_{\left[\left[\right]\right]}RT < TRel > \text{TargetTerm} T \rightsquigarrow (\left[\left[\right]\right], T) \in TRel^*$
using indRelRTPO-to-TRel(2)[where $TRel = TRel$] trans-closure-of-TRel-refl-cond
by simp
moreover assume operational-complete ($TRel^*$)
\[ \land \text{weak-reduction-simulation } (TRel^*) \]
hence weak-reduction-simulation (indRelRTPO $TRel$) (STCal Source Target)
using OCom-iff-indRelRTPO-is-weak-reduction-simulation[where TRel=TRel]
by simp
ultimately show ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
  ∧ (∀ T1 T2. (T1, T2) ∈ TRel → (TargetTerm T1, TargetTerm T2) ∈ Rel)
  ∧ (∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel → (T1, T2) ∈ TRel⁺)
  ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel⁺)
  ∧ weak-reduction-simulation Rel (STCal Source Target)
by blast
next
assume ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
  ∧ (∀ T1 T2. (T1, T2) ∈ TRel → (TargetTerm T1, TargetTerm T2) ∈ Rel)
  ∧ (∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel → (T1, T2) ∈ TRel⁺)
  ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel⁺)
  ∧ weak-reduction-simulation Rel (STCal Source Target)
from this obtain Rel where A1: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
  and A2: ∀ T1 T2. (T1, T2) ∈ TRel → (TargetTerm T1, TargetTerm T2) ∈ Rel
  and A3: ∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel → (T1, T2) ∈ TRel⁺
  and A4: ∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel⁺
  and A5: weak-reduction-simulation Rel (STCal Source Target)
by blast
from A1 A4 A5 have operational-complete (TRel⁺)
  using weak-reduction-simulation-impl-OWCom[where Rel=Rel and TRel=TRel]
  by simp
moreover from A2 A3 A5 have weak-reduction-simulation (TRel⁺) Target
  using rel-with-target-implC-TRel-is-weak-reduction-simulation[where Rel=Rel and
  TRel=TRel]
  by simp
ultimately show operational-complete (TRel⁺)
  ∧ weak-reduction-simulation (TRel⁺) Target
  by simp
qed

An encoding is strong operational complete modulo a strong simulation on target terms TRel if there is a relation, like indRelRTPO, that relates at least all source terms to their literal translations, includes TRel, and is a strong simulation.

lemma (in encoding) strong-reduction-simulation-impl-SOCom:
  fixes Rel :: (′procS, ′procT) Proc × (′procS, ′procT) Proc set
  and TRel :: (′procT × ′procT) set
  assumes A1: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
    and A2: ∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel⁺
    and A3: strong-reduction-simulation Rel (STCal Source Target)
  shows strongly-operational-complete (TRel⁺)

proof clarify
fix S S'
from A1 have (SourceTerm S, TargetTerm ([S])) ∈ Rel
  by simp
moreover assume S → Source S'
  hence SourceTerm S → STCal Source Target (SourceTerm S')
  by (simp add: STCal-step(1))
ultimately obtain Q' where A5: TargetTerm ([S]) → (STCal Source Target) Q'
  and A6: (SourceTerm S', Q') ∈ Rel
    using A3
    by blast
from A5 obtain T where A7: T ∈ T Q' and A8: [S] → Target T
  by (auto simp add: STCal-step(2))
from A2 A6 A7 have ([S'], T) ∈ TRel⁺
  by simp
with A8 show ∃ T. [S] → Target T ∧ ([S'], T) ∈ TRel⁺
  by blast
qed
lemma (in encoding) SOCom-iff-indRelRTPO-is-strong-reduction-simulation:
  fixes TRel :: (′proc T × ′proc T) set
  shows (strongly-operational-complete (TRel+))
    ∧ strong-reduction-simulation (TRel+) Target
  = strong-reduction-simulation (indRelRTPO TRel) (STCal Source Target)
proof (rule iffI, erule conjE)
  assume soc: strongly-operational-complete (TRel+)
  and sim: strong-reduction-simulation (TRel+) Target
  show strong-reduction-simulation (indRelRTPO TRel) (STCal Source Target)
proof clarify
  fix P Q P′
  assume P ≲ ′[TRel] Q and P ↦ (STCal Source Target) P′
  thus ∃ Q′. Q ↦ (STCal Source Target) Q′ ∧ P′ ≲ ′[TRel] Q′
proof (induct arbitrary: P′)
  case (encR S)
  assume SourceTerm S ↦ (STCal Source Target) P′
  from this obtain S′ where A1: S′ ∈ S P′ and A2: S ↦ Source S′
    by (auto simp add: STCal-step(1))
  from soc A2 obtain T where A3: [S] ↦ Target T and A4: ([S′], T) ∈ TRel+
    by blast
  from A3 have TargetTerm ([S]) ↦ (STCal Source Target) (TargetTerm T)
    by (simp add: STCal-step(2))
  moreover have P′ ≲ ′[TRel] TargetTerm T
proof –
  from A4 have [S′] = T ∨ ([S′], T) ∈ TRel+
    using rtrancl-eq-or-rtrancl[of [S′] T TRel]
    by blast
  moreover from A1 have A5: P′ ≲ ′[TRel] TargetTerm ([S′])
    by (simp add: indRelRTPO.encR)
  hence [S′] = T ⇒ P′ ≲ ′[TRel] TargetTerm T
    by simp
  moreover have ([S′], T) ∈ TRel+ ⇒ P′ ≲ ′[TRel] TargetTerm T
proof –
  assume ([S′], T) ∈ TRel+
  hence TargetTerm ([S′]) ≲ ′[TRel] TargetTerm T
proof induct
  fix TQ
  assume ([S′], TQ) ∈ TRel
  thus TargetTerm ([S′]) ≲ ′[TRel] TargetTerm TQ
    by (rule indRelRTPO.target)
next
case (step TQ TR)
  assume TargetTerm ([S′]) ≲ ′[TRel] TargetTerm TQ
  moreover assume (TQ, TR) ∈ TRel
  hence TargetTerm TQ ≲ ′[TRel] TargetTerm TR
    by (rule indRelRTPO.target)
  ultimately show TargetTerm ([S′]) ≲ ′[TRel] TargetTerm TR
    by (rule indRelRTPO.trans)
  qed
  with A5 show P′ ≲ ′[TRel] TargetTerm T
    by (rule indRelRTPO.trans)
  qed
  ultimately show P′ ≲ ′[TRel] TargetTerm T
    by blast
  qed
next
case (source S)

then obtain $S'$ where $B_1: S' \in S P'$
by (auto simp add: STCal-step(1))
hence $P' \leq \llbracket RT \rrbracket_{Trel} > P'$
by (simp add: indRelRTPO.to-TRel)
with source show $\exists Q'. \ SourceTerm S \mapsto (STCal \ Source \ Target) \ Q' \land P' \leq \llbracket RT \rrbracket_{Trel} > Q'$
by blast

next

case (target $T_1 \ T_2$)
assume TargetTerm $T_1 \mapsto (STCal \ Source \ Target) \ P'$
from this obtain $T_1'$ where $C_1: T_1' \in T P'$ and $C_2: T_1 \mapsto \ Target \ T_1'$
by (auto simp add: STCal-step(2))
assume $(T_1, T_2) \in TRel$
hence $(T_1, T_2) \in \ TRel^+$
by simp
with $C_2$ sim obtain $T_2'$ where $C_3: T_2 \mapsto \ Target \ T_2'$ and $C_4: (T_1', T_2') \in \ TRel^+$
by blast
from $C_3$ have TargetTerm $T_2 \mapsto (STCal \ Source \ Target) (TargetTerm \ T_2')$
by (simp add: STCal-step(2))
moreover from $C_4$ have TargetTerm $T_1' \leq \llbracket RT \rrbracket_{Trel} > TargetTerm \ T_2'$
proof induct
fix $T_2'$
assume $(T_1', T_2') \in \ TRel$
thus TargetTerm $T_1' \leq \llbracket RT \rrbracket_{Trel} > TargetTerm \ T_2'$
by (rule indRelRTPO.target)

next

case (step $TQ \ TR$)
assume TargetTerm $T_1' \leq \llbracket RT \rrbracket_{Trel} > TargetTerm \ TQ$
moreover assume $(TQ, TR) \in TRel$
hence TargetTerm $TQ \leq \llbracket RT \rrbracket_{Trel} > TargetTerm \ TR$
by (rule indRelRTPO.target)
ultimately show TargetTerm $T_1' \leq \llbracket RT \rrbracket_{Trel} > TargetTerm \ TR$
by (rule indRelRTPO.trans)

qed

with $C_1$ have $P' \leq \llbracket RT \rrbracket_{Trel} > TargetTerm \ T_2'$
by simp
ultimately show $\exists Q'. \ TargetTerm \ T_2 \mapsto (STCal \ Source \ Target) \ Q' \land P' \leq \llbracket RT \rrbracket_{Trel} > Q'$
by blast

next

case (trans $P \ Q \ R$)
assume $P \mapsto (STCal \ Source \ Target) \ P'$
and $\forall P', P \mapsto (STCal \ Source \ Target) \ P'$
implies $\exists Q'. \ Q \mapsto (STCal \ Source \ Target) \ Q' \land P' \leq \llbracket RT \rrbracket_{Trel} > Q'$
from this obtain $Q'$ where $D_1: Q \mapsto (STCal \ Source \ Target) \ Q'$
and $D_2: P' \leq \llbracket RT \rrbracket_{Trel} > Q'$
by blast
assume $\forall Q'. \ Q \mapsto (STCal \ Source \ Target) \ Q'$
implies $\exists R'. \ R \mapsto (STCal \ Source \ Target) \ R' \land Q' \leq \llbracket RT \rrbracket_{Trel} > R'$
with $D_1$ obtain $R'$ where $D_3: R \mapsto (STCal \ Source \ Target) \ R'$
and $D_4: Q' \leq \llbracket RT \rrbracket_{Trel} > R'$
by blast
from $D_2 \ D_4$ have $P' \leq \llbracket RT \rrbracket_{Trel} > R'$
by (rule indRelRTPO.trans)
with $D_3$ show $\exists R'. \ R \mapsto (STCal \ Source \ Target) \ R' \land P' \leq \llbracket RT \rrbracket_{Trel} > R'$
by blast

qed

next

have $\forall S. \ SourceTerm S \leq \llbracket RT \rrbracket_{Trel} \ TargetTerm ([S])$
by (simp add: indRelRTPO.encR)
moreover have $\forall S \ T. \ SourceTerm S \leq \llbracket RT \rrbracket_{Trel} > TargetTerm \ T \mapsto (\llbracket S \rrbracket, T) \in \ TRel^*$
using indRelRTPO-to-TRel(2) [where $TRel = TRel$] trans-closure-of-TRel-refl-cond

164
by simp
moreover assume sim: strong-reduction-simulation (indRelRTPO TRel) (STCal Source Target)
ultimately have strongly-operational-complete (TRel*)
  using strong-reduction-simulation-impl-SOCom[where Rel=indRelRTPO TRel and TRel=TRel]
by simp
moreover from sim have strong-reduction-simulation (TRel+) Target
  using indRelRTPO-impl-TRel-is-strong-reduction-simulation[where TRel=TRel]
by simp
ultimately show strongly-operational-complete (TRel*)
  ∧ strong-reduction-simulation (TRel+) Target
by simp
qed

lemma (in encoding) SOCom_iff-strong-reduction-simulation:
  fixes TRel :: ('procT × 'procT) set
  shows (strongly-operational-complete (TRel*)
  ∧ strong-reduction-simulation (TRel+) Target
  = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ indRelRTPO TRel
  ∧ (∀ T1 T2. (T1, T2) ∈ TRel → TargetTerm T1 ⪯ TRel> TargetTerm T2 ∈ Rel)
  ∧ (∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel → (T1, T2) ∈ TRel+)
  ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel*)
  ∧ strong-reduction-simulation Rel (STCal Source Target))
proof (rule iffI, erule conjE)
  have v S. (SourceTerm S, TargetTerm ([S])) ∈ indRelRTPO TRel
  by (simp add: indRelRTPO_encR)
  moreover have ∀ T1 T2. (T1, T2) ∈ TRel → TargetTerm T1 ⪯ TRel> TargetTerm T2 ∈ Rel
  by (simp add: indRelRTPO_target)
  moreover have ∀ T1 T2. TargetTerm T1 ⪯ TRel> TargetTerm T2 ∈ Rel → (T1, T2) ∈ TRel+
  using indRelRTPO-to-TRel[2][where TRel=TRel]
  by simp
  moreover have ∀ S T. SourceTerm S ⪯ TRel> TargetTerm T → ([S], T) ∈ TRel*
  using indRelRTPO-to-TRel[2][where TRel=TRel] trans-closure-of-TRel-refl-cond
  by simp
moreover assume strongly-operational-complete (TRel*)
  and strong-reduction-simulation (TRel+) Target
hence strong-reduction-simulation (indRelRTPO TRel) (STCal Source Target)
  using SOCom_iff-indRelRTPO-is-strong-reduction-simulation[where TRel=TRel]
  by simp
ultimately show ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
  ∧ (∀ T1 T2. (T1, T2) ∈ TRel → (TargetTerm T1, TargetTerm T2) ∈ Rel)
  ∧ (∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel → (T1, T2) ∈ TRel+)
  ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel*)
  ∧ strong-reduction-simulation Rel (STCal Source Target)
  by blast
next
  assume ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
  ∧ (∀ T1 T2. (T1, T2) ∈ TRel → (TargetTerm T1, TargetTerm T2) ∈ Rel)
  ∧ (∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel → (T1, T2) ∈ TRel+)
  ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel*)
  ∧ strong-reduction-simulation Rel (STCal Source Target)
  from this obtain Rel where A1: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
  and A2: ∀ T1 T2. (T1, T2) ∈ TRel → (TargetTerm T1, TargetTerm T2) ∈ Rel
  and A3: ∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel → (T1, T2) ∈ TRel+
  and A4: ∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel*
  and A5: strong-reduction-simulation Rel (STCal Source Target)
  by blast
from A1 A4 A5 have strongly-operational-complete (TRel*)
  using strong-reduction-simulation-impl-SOCom[where Rel=Rel and TRel=TRel]
by simp
moreover from A2 A3 A5 have strong-reduction-simulation (TRel+) Target
  using rel-with-target-impl-transC-TRel-is-strong-reduction-simulation[where Rel=Rel and
lemma (in encoding) target-relation-from-source-target-relation:
  fixes Rel :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) set
  assumes stre: ∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel
                   → (TargetTerm ([S]), TargetTerm T) ∈ Rel\(^{−}\)
  shows ∀ TRel. (∃ T1 T2. (T1, T2) ∈ TRel → (TargetTerm T1, TargetTerm T2) ∈ Rel)
               ∧ (∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel → (T1, T2) ∈ TRel\(^{+}\))
               ∧ (∃ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel\(^{∗}\))

proof
  define TRel where TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
  from TRel-def have ∀ T1 T2. (T1, T2) ∈ TRel → (TargetTerm T1, TargetTerm T2) ∈ Rel\(^{+}\)
           by simp
  moreover from TRel-def
  have ∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel → (T1, T2) ∈ TRel\(^{+}\)
           by blast
  moreover from stre TRel-def
  have ∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel\(^{∗}\)
           by blast
  ultimately show ?thesis
    by blast
qed

lemma (in encoding) SOCom-modulo-TRel-iff-strong-reduction-simulation:
  shows (∃ TRel. strongly-operational-complete (TRel\(^{+}\))
            ∧ strong-reduction-simulation (TRel\(^{+}\)) Target)
        = (∃ TRel. (∃ S T. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
           ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → (TargetTerm ([S]), TargetTerm T) ∈ Rel\(^{−}\))
           ∧ strong-reduction-simulation Rel (STCal Source Target))

proof (rule iffI)
  assume ∃ TRel. strongly-operational-complete (TRel\(^{+}\))
              ∧ strong-reduction-simulation (TRel\(^{+}\)) Target
  from this obtain TRel where strongly-operational-complete (TRel\(^{+}\))
       and strong-reduction-simulation (TRel\(^{+}\)) Target
       by blast
  hence strong-reduction-simulation (indRelRTPO TRel) (STCal Source Target)
    using SOCom-iff-indRelRTPO-is-strong-reduction-simulation[where TRel=TRel]
    by simp
  moreover have ∀ S. SourceTerm S \preceq\{\} RT< TRel> TargetTerm ([S])
        by (simp add: indRelRTPO.encR)
  moreover have ∀ S T. SourceTerm S \preceq\{\} RT< TRel TargetTerm T
        → (TargetTerm ([S]), TargetTerm T) ∈ (indRelRTPO TRel\(^{=}\))
        using indRelRTPO-relates-source-target[where TRel=TRel]
        by simp
  ultimately show ∃ TRel. (∃ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
         ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → (TargetTerm ([S]), TargetTerm T) ∈ Rel\(^{−}\))
         ∧ strong-reduction-simulation Rel (STCal Source Target)
         by blast
next
  assume ∃ TRel. (∃ S T. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
         ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → (TargetTerm ([S]), TargetTerm T) ∈ Rel\(^{−}\))
         ∧ strong-reduction-simulation Rel (STCal Source Target)
  from this obtain Rel where A1: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
        and A2: (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel
        → (TargetTerm ([S]), TargetTerm T) ∈ Rel\(^{−}\))
and A3: \(\text{strong-reduction-simulation} \ Rel \ (\text{STCal} \ Source \ Target)\)

by \(\text{blast}\)
from A2 obtain \(\text{TRel} \ where \ \forall \ T1. \ T2. \ (T1, \ T2) \in \text{TRel} \rightarrow (\text{TargetTerm} T1, \ \text{TargetTerm} T2) \in \text{Rel}\)
and \(\forall \ T1. \ T2. \ (\text{TargetTerm} T1, \ \text{TargetTerm} T2) \in \text{Rel} \rightarrow (T1, \ T2) \in \text{TRel}^+\)
and \(\forall \ S. \ T. \ (\text{SourceTerm} S, \ \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], \ T) \in \text{TRel}^*\)
using \(\text{target-relation-from-source-target-relation}[\text{where} \ \text{Rel}=\text{Rel}]\)
by \(\text{blast}\)
with A1 A3 have \((\text{TRel}^*) \wedge \text{strong-reduction-simulation} \ (\text{TRel}^+) \ \text{Target}\)
using \(\text{SOCom-iff-strong-reduction-simulation}[\text{where} \ \text{TRel}=\text{TRel}]\)
by \(\text{blast}\)
thus \(\exists \ T. \ \text{TRel. \ strongly-operational-complete} \ (\text{TRel}^*)\)
\wedge \(\text{strong-reduction-simulation} \ (\text{TRel}^+) \ \text{Target}\)
by \(\text{blast}\)
\[\text{qed}\]

8.3 Weak Operational Soundness vs Contrasimulation

If the inverse of a relation that includes TRel and relates source terms and their literal translations is a contrasimulation, then the encoding is weakly operational sound.

**lemma (in encoding) weak-reduction-contrasimulation-impl-WOSou:**

fixes \(\text{Rel} \:: \{('\text{procS}', '\text{procT}) \ \text{Proc} \times ('\text{procS}', '\text{procT}) \ \text{Proc}\ \text{set}\)
and \(\text{TRel} \:: \{('\text{procT} \times '\text{procT}) \ \text{set}\)
assumes A1: \(\forall \ S. \ (\text{SourceTerm} S, \ \text{TargetTerm} ([S])) \in \text{Rel}\)
and A2: \(\forall \ S. \ T. \ (\text{SourceTerm} S, \ \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], \ T) \in \text{TRel}^*\)
and A3: \(\text{weak-reduction-contrasimulation} \ (\text{Rel}^{-1}) \ (\text{STCal} \ Source \ Target)\)
shows \(\text{weakly-operational-sound} \ (\text{TRel}^*)\)
proof clarify
fix \(S. \ T\)
from A1 have \((\text{TargetTerm} ([S]), \ \text{SourceTerm} S) \in \text{Rel}^{-1}\)
by \(\text{simp}\)
moreover assume \([S] \rightarrow \text{Targets} \ T\)
hence \(\text{TargetTerm} ([S]) \rightarrow (\text{STCal} \ Source \ Target)^* \ (\text{TargetTerm} T)\)
by \(\text{(simp add: STCal-steps(2))}\)
ultimately obtain \(Q' \ where \ A5: \ \text{SourceTerm} S \rightarrow (\text{STCal} \ Source \ Target)^* \ Q'\)
and A6: \((Q', \ \text{TargetTerm} T) \in \text{Rel}^{-1}\)
using A3
by \(\text{blast}\)
from A5 obtain \(S' \ where \ A7: \ S' \in S \ Q'\) \ and \(A8: \ S \rightarrow \text{Source}* \ S'\)
by \(\text{(auto simp add: STCal-steps(I))}\)
have \(Q' \rightarrow (\text{STCal} \ Source \ Target)^* \ Q'\)
by \(\text{(simp add: steps-refl)}\)
with A6 A3 obtain \(P'' \ where \ A9: \ \text{TargetTerm} T \rightarrow (\text{STCal} \ Source \ Target)^* \ P''\)
and A10: \((P'', \ Q') \in \text{Rel}^{-1}\)
by \(\text{blast}\)
from A9 obtain \(T' \ where \ A11: \ T' \in T \ P''\) \ and \(A12: \ T \rightarrow \text{Targets} \ T'\)
by \(\text{(auto simp add: STCal-steps(2))}\)
from A10 have \((Q', \ P'') \in \text{Rel}\)
by \(\text{induct}\)
with A2 A7 A11 have \([S'], \ T') \in \text{TRel}^*\)
by \(\text{simp}\)
with A8 A12 show \(\exists \ S', \ T'. \ S \rightarrow \text{Source}* \ S' \wedge T \rightarrow \text{Target}* \ T' \wedge ([S'], \ T') \in \text{TRel}^*\)
by \(\text{blast}\)
\[\text{qed}\]

8.4 (Strong) Operational Soundness vs (Strong) Simulation

An encoding is operational sound modulo a relation TRel whose inverse is a weak reduction simulation on target terms iff there is a relation, like \(\text{indRelRTPo}\), that relates at least all source terms to their literal translations, includes TRel, and whose inverse is a weak simulation.
lemma (in encoding) weak-reduction-simulation-impl-OSou:
  fixes Rel :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) set
  and TRel :: ('procT × 'procT) set
  assumes A1: ∀S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
  and A2: ∀S T. (SourceTerm S, TargetTerm T) ∈ Rel −→ ([S], T) ∈ TRel+
  and A3: weak-reduction-simulation (Rel⁻¹) (STCal Source Target)
  shows operational-sound (TRel⁺)
proof clarify
  fix S T
  from A1 have (TargetTerm ([S]), SourceTerm S) ∈ Rel⁻¹
    by simp
  moreover assume [S] −→ Target* T
  hence TargetTerm ([S]) −→ (STCal Source Target)* (TargetTerm T)
    by (simp add: STCal-steps(2))
  ultimately obtain Q’ where A5: SourceTerm S −→ (STCal Source Target)* Q’
    and A6: (TargetTerm T, Q’) ∈ Rel⁻¹
      using A3
    by blast
  from A5 obtain S’ where A7: S’ ∈ S Q’ and A8: S −→ Source* S’
    by (auto simp add: STCal-steps(1))
  from A6 have (Q’, TargetTerm T) ∈ Rel
    by induct
  with A2 A7 have ([S’], T) ∈ TRel⁺
    by simp
  with A8 show ∃S’. S −→ Source* S’ ∧ ([S’], T) ∈ TRel⁺
    by blast
qed

lemma (in encoding) OSou-iff-inverse-of-indRelRTPO-is-weak-reduction-simulation:
  fixes TRel :: ('procT × 'procT) set
  shows (operational-sound (TRel⁺)
    ∧ weak-reduction-simulation ((TRel⁺)⁻¹) Target
    = weak-reduction-simulation ((indRelRTPO TRel)⁻¹) (STCal Source Target)
proof (rule iffI, erule conjE)
  assume os: operational-sound (TRel⁺)
  and sim: weak-reduction-simulation ((TRel⁺)⁻¹) Target
  show weak-reduction-simulation ((indRelRTPO TRel)⁻¹) (STCal Source Target)
proof clarify
  fix P Q P’
  assume Q ≤• [1]RT < TRel> P and P −→ (STCal Source Target)* P’
  thus ∃Q’. Q −→ (STCal Source Target)* Q’ ∧ (P’, Q’) ∈ (indRelRTPO TRel)⁻¹
proof (induct arbitrary: P')
  case (encR S)
  assume TargetTerm ([S]) −→ (STCal Source Target)* P'
  from this obtain T where A1: T ∈ T P’ and A2: [S] −→ Target* T
    by (auto simp add: STCal-steps(2))
  from os A2 obtain S’ where A3: S −→ Source* S’ and A4: ([S’], T) ∈ TRel⁺
    by blast
  from A3 have SourceTerm S −→ (STCal Source Target)* (SourceTerm S’)
    by (simp add: STCal-steps(1))
  moreover have SourceTerm S’ ≤• [1]RT < TRel> P'
  proof
    from A4 have [S’] = T ∨ ([S’], T) ∈ TRel⁺
      using rtrancl-eq-or-rtrancl[of [S’] T TRel]
      by blast
    moreover have A5: SourceTerm S’ ≤• [1]RT < TRel> TargetTerm ([S’])
      by (simp add: indRelRTPO.encR)
    with A1 have [S’] = T −→ SourceTerm S’ ≤• [1]RT < TRel> P'
      by simp
    moreover have ([S’], T) ∈ TRel⁺ −→ SourceTerm S’ ≤• [1]RT < TRel> P'
    proof –
assume \([S'], T\) \(\in\) \(\mathit{TRel}^+\)

hence \(\mathit{TargetTerm} ([S']) \leq [\cdot] \mathit{RT} \mathit{\langle TRel \rangle} \mathit{TargetTerm} \mathit{T}\)

by \((\text{rule transitive-closure-of-TRel-to-indRelRTPO})\)

with \(A5\) have \(\mathit{SourceTerm} S' \leq [\cdot] \mathit{RT} \mathit{\langle TRel \rangle} \mathit{TargetTerm} \mathit{T}\)

by \((\text{rule indRelRTPO.trans})\)

with \(A1\) show \(\mathit{SourceTerm} S' \leq [\cdot] \mathit{RT} \mathit{\langle TRel \rangle} \mathit{P'}\)

by \(\text{simp}\)

qed

ultimately show \(\mathit{SourceTerm} S' \leq [\cdot] \mathit{RT} \mathit{\langle TRel \rangle} \mathit{P'}\)

by \(\text{blast}\)

next

case \((\text{source} \; S)\)

then obtain \(S'\) where \(B1: \; S' \in S \; P'\)

by \((\text{auto simp add: STCal-steps(1)})\)

hence \((P', P') \in \langle \text{indRelRTPO TRel} \rangle^{-1}\)

by \((\text{simp add: indRelRTPO.source})\)

with \(\text{source}\)

show \(\exists \; Q'. \; \mathit{SourceTerm} \; S \mathbin{\mathit{\langle STCal \; \text{Source \; Target} \rangle}^*} \; Q' \wedge (P', Q') \in \langle \text{indRelRTPO \; TRel} \rangle^{-1}\)

by \(\text{blast}\)

next

case \((\text{target} \; T1 \; T2)\)

assume \(\mathit{TargetTerm} \; T2 \mathbin{\mathit{\langle STCal \; \text{Source \; Target} \rangle}^*} \; P'\)

from this obtain \(T2'\) where \(C1: \; T2' \in T \; P'\) and \(C2: \; T2 \mathbin{\mathit{\langle \text{Target} \; \text{Target} \rangle}^*} \; T2'\)

by \((\text{auto simp add: STCal-steps(2)})\)

assume \((T1, T2) \in \mathit{TRel}\)

hence \((T2', T1) \in \langle \mathit{TRel}^+ \rangle^{-1}\)

by \(\text{simp}\)

with \(C2\) sim obtain \(T1'\) where \(C3: \; T1 \mathbin{\mathit{\langle \text{Target} \; \text{Target} \rangle}^*} \; T1'\) and \(C4: \; (T2', T1') \in \langle \mathit{TRel}^+ \rangle^{-1}\)

by \(\text{blast}\)

from \(C3\) have \(\mathit{TargetTerm} \; T1 \mathbin{\mathit{\langle STCal \; \text{Source \; Target} \rangle}^*} \; (\mathit{TargetTerm} \; T1')\)

by \((\text{simp add: STCal-steps(2)})\)

moreover from \(C4\) have \((T1', T2') \in \mathit{TRel}^+\)

by \(\text{induct}\)

hence \(\mathit{TargetTerm} \; T1' \leq [\cdot] \mathit{RT} \mathit{\langle TRel \rangle} \mathit{TargetTerm} \; T2'\)

by \((\text{rule transitive-closure-of-TRel-to-indRelRTPO})\)

with \(C1\) have \((P', \mathit{TargetTerm} \; T1') \in \langle \text{indRelRTPO \; TRel} \rangle^{-1}\)

by \(\text{simp}\)

ultimately

show \(\exists \; Q'. \; \mathit{TargetTerm} \; T1 \mathbin{\mathit{\langle STCal \; \text{Source \; Target} \rangle}^*} \; Q' \wedge (P', Q') \in \langle \text{indRelRTPO \; TRel} \rangle^{-1}\)

by \(\text{blast}\)

next

case \((\text{trans} \; P \; Q \; R \; R')\)

assume \(R \mathbin{\mathit{\langle STCal \; \text{Source \; Target} \rangle}^*} \; R'\)

and \(\wedge \; R'. \; R \mathbin{\mathit{\langle STCal \; \text{Source \; Target} \rangle}^*} \; R'\)

\(\implies \exists \; Q'. \; Q \mathbin{\mathit{\langle STCal \; \text{Source \; Target} \rangle}^*} \; Q' \wedge (R', Q') \in \langle \text{indRelRTPO \; TRel} \rangle^{-1}\)

from this obtain \(Q'\) where \(D1: \; Q \mathbin{\mathit{\langle STCal \; \text{Source \; Target} \rangle}^*} \; Q'\)

and \(D2: \; (R', Q') \in \langle \text{indRelRTPO \; TRel} \rangle^{-1}\)

by \(\text{blast}\)

assume \(\wedge \; Q'. \; Q \mathbin{\mathit{\langle STCal \; \text{Source \; Target} \rangle}^*} \; Q'\)

\(\implies \exists \; P'. \; P \mathbin{\mathit{\langle STCal \; \text{Source \; Target} \rangle}^*} \; P' \wedge (Q', P') \in \langle \text{indRelRTPO \; TRel} \rangle^{-1}\)

with \(D1\) obtain \(P'\) where \(D3: \; P \mathbin{\mathit{\langle STCal \; \text{Source \; Target} \rangle}^*} \; P'\)

and \(D4: \; (Q', P') \in \langle \text{indRelRTPO \; TRel} \rangle^{-1}\)

by \(\text{blast}\)

from \(D4 \; D2\) have \((R', P') \in \langle \text{indRelRTPO \; TRel} \rangle^{-1}\)

by \((\text{simp add: indRelRTPO.trans[where P=P' and Q=Q' and R=R']})\)
with D̂3 show \( \exists P'. P \equiv (\textit{STCal Source Target})^* P' \land (\textit{R}', P') \in (\textit{indRelRTPO TRel})^{-1} \)
by blast
qed
qed

next

have \( \forall S. \textit{SourceTerm} S \leq [\cdot] \textit{RT} < \textit{TRel} \rightarrow \textit{TargetTerm} ([S]) \)
by (simp add: \textit{indRelRTPO, encR})

moreover have \( \forall S T. \textit{SourceTerm} S \leq [\cdot] \textit{RT} < \textit{TRel} \rightarrow \textit{TargetTerm} T \rightarrow ([S], T) \in \textit{TRel}^* \)
using \textit{indRelRTPO-to-TRel(2)[where TRel=\textit{TRel}] trans-closure-of-\textit{TRel-refl-cond}}
by simp

moreover assume sim: weak-reduction-simulation \(((\textit{indRelRTPO TRel})^{-1}) (\textit{STCal Source Target})

ultimately have operational-sound (\textit{TRel}^*)
using weak-reduction-simulation-impl-\textit{OSou}[where Rel=\textit{indRelRTPO TRel} and \textit{TRel}=\textit{TRel}]
by simp

moreover from sim have weak-reduction-simulation \(((\textit{TRel}^*)^{-1}) \textit{Target}
using \textit{indRelRTPO-impl-\textit{TRel-is-weak-reduction-simulation-rev}[where TRel=\textit{TRel}]
by simp

ultimately show operational-sound (\textit{TRel}^*) \land weak-reduction-simulation \(((\textit{TRel}^*)^{-1}) \textit{Target}
by simp

qed

lemma (in encoding) \textit{OSou-if-weak-reduction-simulation}:

fixes \textit{TRel}: \('', proCT \times 'proCT) \textit{set}

shows (operational-sound (\textit{TRel}^*)
\land weak-reduction-simulation \(((\textit{TRel}^*)^{-1}) \textit{Target}
=
\exists \textit{Rel}. \forall S. \textit{SourceTerm} S, \textit{TargetTerm} ([S]) \in \textit{indRelRTPO TRel}
\land (\forall T1 T2. (T1, T2) \in \textit{TRel} \rightarrow (\textit{TargetTerm T1, TargetTerm T2}) \in \textit{Rel})
\land (\forall T1 T2. (T1, T2) \in \textit{TRel} \rightarrow (T1, T2) \in \textit{\textit{TRel}^*})
\land (\forall S T. \textit{SourceTerm} S, \textit{TargetTerm} T \in \textit{Rel} \rightarrow ([S], T) \in \textit{\textit{TRel}^*})
\land weak-reduction-simulation (\textit{Rel}^{-1}) (\textit{STCal Source Target})

proof (rule \textit{iffI, erule conjE})

have \( \forall S. \textit{SourceTerm} S, \textit{TargetTerm} ([S]) \in \textit{indRelRTPO TRel}
by (simp add: \textit{indRelRTPO, encR})

moreover have \( \forall T1 T2. (T1, T2) \in \textit{TRel} \rightarrow \textit{TargetTerm T1} \leq [\cdot] \textit{RT} < \textit{TRel} \rightarrow \textit{TargetTerm T2}
by (simp add: \textit{indRelRTPO-target})

moreover have \( \forall T1 T2. \textit{TargetTerm} T1 \leq [\cdot] \textit{RT} < \textit{TRel} \rightarrow \textit{TargetTerm T2} \rightarrow (T1, T2) \in \textit{\textit{TRel}^*}
using \textit{indRelRTPO-to-TRel(4)[where TRel=\textit{TRel}]}
by simp

moreover have \( \forall S T. \textit{SourceTerm} S \leq [\cdot] \textit{RT} < \textit{TRel} \rightarrow \textit{TargetTerm T} \rightarrow ([S], T) \in \textit{\textit{TRel}^*}
using \textit{indRelRTPO-to-TRel(2)[where TRel=\textit{TRel}] trans-closure-of-\textit{TRel-refl-cond}}
by simp

moreover assume operational-sound (\textit{TRel}^*)
and weak-reduction-simulation \(((\textit{TRel}^*)^{-1}) \textit{Target}
using \textit{OSou-if-inverse-of-indRelRTPO-is-weak-reduction-simulation}[where TRel=\textit{TRel}]
by simp

ultimately show \( \exists \textit{Rel}. \forall S. \textit{SourceTerm} S, \textit{TargetTerm} ([S]) \in \textit{Rel}
\land (\forall T1 T2. (T1, T2) \in \textit{TRel} \rightarrow (\textit{TargetTerm T1, TargetTerm T2}) \in \textit{Rel})
\land (\forall T1 T2. (T1, T2) \in \textit{TRel} \rightarrow (T1, T2) \in \textit{\textit{TRel}^*})
\land (\forall S T. \textit{SourceTerm} S, \textit{TargetTerm} T \in \textit{Rel} \rightarrow ([S], T) \in \textit{\textit{TRel}^*})
\land weak-reduction-simulation (\textit{Rel}^{-1}) (\textit{STCal Source Target})

by blast

next

assume \( \exists \textit{Rel}. \forall S. \textit{SourceTerm} S, \textit{TargetTerm} ([S]) \in \textit{Rel}
\land (\forall T1 T2. (T1, T2) \in \textit{TRel} \rightarrow (\textit{TargetTerm T1, TargetTerm T2}) \in \textit{Rel})
\land (\forall T1 T2. (T1, T2) \in \textit{TRel} \rightarrow (T1, T2) \in \textit{\textit{TRel}^*})
\land (\forall S T. \textit{SourceTerm} S, \textit{TargetTerm} T \in \textit{Rel} \rightarrow ([S], T) \in \textit{\textit{TRel}^*})
\land weak-reduction-simulation (\textit{Rel}^{-1}) (\textit{STCal Source Target})

from this obtain \textit{Rel}[where A1: \forall S. \textit{SourceTerm} S, \textit{TargetTerm} ([S]) \in \textit{Rel}
and A2: \forall T1 T2. (T1, T2) \in \textit{TRel} \rightarrow (\textit{TargetTerm T1, TargetTerm T2}) \in \textit{Rel}
and A3: \( \forall T_1 T_2. (\text{TargetTerm} T_1, \text{TargetTerm} T_2) \in \text{Rel} \rightarrow (T_1, T_2) \in \text{TRel}^+ \)

and A4: \( \forall S T. (\text{SourceTerm} S, \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}^* \)

and A5: weak-reduction-simulation \( (\text{Rel}^{-1}) \) \( (\text{STCal Source Target}) \)
   by blast

from A1 A4 A5 have operational-sound \( (\text{TRel}^*) \)
   using weak-reduction-simulation-impl-OSou[where Rel=\text{Rel} and TRel=\text{TRel}]
   by simp

moreover from A2 A3 A5 have weak-reduction-simulation \( ((\text{TRel}^*)^{-1}) \) \( \text{Target} \)
   using rel-with-target-impl-transC-TRel-is-weak-reduction-simulation-rev[where Rel=\text{Rel} and TRel=\text{TRel}]
   by simp

ultimately show operational-sound \( (\text{TRel}^*) \) \( \land \) weak-reduction-simulation \( ((\text{TRel}^*)^{-1}) \) \( \text{Target} \)
   by simp

qed

An encoding is strongly operational sound modulo a relation \( \text{TRel} \) whose inverse is a strong reduction simulation on target terms iff there is a relation, like \( \text{indRelRTPO} \), that relates at least all source terms to their literal translations, includes \( \text{TRel} \), and whose inverse is a strong simulation.

**Lemma (in encoding)** strong-reduction-simulation-impl-SOSou:

fixes \( \text{Rel} :: (\{\text{procS}, \text{procT}\} \ \text{Proc} \times \{\text{procS}, \text{procT}\} \ \text{Proc}) \ \text{set} \)

and \( \text{TRel} :: (\{\text{procT} \times \text{procT}\} \ \text{set} \)

assumes A1: \( \forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel} \)

and A2: \( \forall S T. (\text{SourceTerm} S, \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}^* \)

and A3: strongly-operational-sound \( (\text{Rel}^{-1}) \) \( (\text{STCal Source Target}) \)

shows strongly-operational-sound \( (\text{TRel}^*) \)

**Proof** clarify

fix \( S T \)

from A1 have \( (\text{TargetTerm} ([S]), \text{SourceTerm} S) \in \text{Rel}^{-1} \)
   by simp

moreover assume \( [S] \rightarrow \text{Target} T \)

hence \( \text{TargetTerm} ([S]) \rightarrow (\text{STCal Source Target}) (\text{TargetTerm} T) \)
   by (simp add: STCal-step(2))

ultimately obtain \( Q' \) where A5: \( \text{SourceTerm} S \rightarrow (\text{STCal Source Target}) Q' \)

and A6: \( (\text{TargetTerm} T, Q') \in \text{Rel}^{-1} \)

   using A3
   by blast

from A5 obtain \( S' \) where A7: \( S' \in S Q' \) and A8: \( S \rightarrow \text{Source} S' \)
   by (auto simp add: STCal-step(1))

from A6 have \( (Q', \text{TargetTerm} T) \in \text{Rel} \)
   by induct

with A2 A7 have \( ([S], T) \in \text{TRel}^* \)
   by simp

with A8 show \( \exists S'. S \rightarrow \text{Source} S' \land ([S'], T) \in \text{TRel}^* \)
   by blast

qed

**Lemma (in encoding)** SOSou-iff-inverse-of-indRelRTPO-is-strong-reduction-simulation:

fixes \( \text{TRel} :: (\text{procT} \times \text{procT}) \ \text{set} \)

shows \( (\text{strongly-operational-sound} (\text{TRel}^*)) \land \text{strong-reduction-simulation} ((\text{TRel}^*)^{-1}) \ \text{Target} \)
   \( = \) \( \text{strong-reduction-simulation} ((\text{indRelRTPO TRel})^{-1}) \) \( (\text{STCal Source Target}) \)

**Proof** (rule iffI,erule conjE)

assume os: strongly-operational-sound \( (\text{TRel}^*) \)

and sim: strongly-reduction-simulation \( ((\text{TRel}^*)^{-1}) \) \( \text{Target} \)

show strongly-reduction-simulation \( ((\text{indRelRTPO TRel})^{-1}) \) \( (\text{STCal Source Target}) \)

**Proof** clarify

fix \( P Q P' \)

assume \( Q \leq_{RT}^* \text{TRel} > P \)

moreover assume \( P \rightarrow (\text{STCal Source Target}) P' \)

ultimately
show \( \exists Q'. \ Q \longrightarrow (\text{STCal Source Target}) \ Q' \land (P', Q') \in \text{(indRelRTPO TRel)}^{-1} \)

proof
(induct arbitrary: \( P' \))
case (\text{encR} S)
assume TargetTerm \( ([S]) \longrightarrow (\text{STCal Source Target}) \ P' \)
from this obtain \( T \) where \( A1: \ T \in T \ P' \) and \( A2: \ [S] \longrightarrow \text{Target Term} \)
by (auto simp add: STCal-step(2))
from as \( \text{A2} \) obtain \( S' \) where \( A3: \ S \longrightarrow S' \) and \( A4: \ ([S'], T) \in \text{TRel}^+ \)
by blast
from \( \text{A3} \) have \( \text{SourceTerm} S \longrightarrow (\text{STCal Source Target})(\text{SourceTerm} S') \)
by (simp add: STCal-step(I))
moreover have \( \text{SourceTerm} S' \lesssim [\cdot]\text{RT}<\text{TRel}> P' \)
proof
\( \text{from} \ A4 \) have \( [S'] = T \lor ([S'], T) \in \text{TRel}^+ \)
using rtrancl-eq-or-trancl[of \( [S'] \) \( T \) \( \text{TRel} \)]
by blast
moreover have \( A5: \ \text{SourceTerm} S' \lesssim [\cdot]RT\lesssim RT<\text{TRel}> \text{Target Term} ([S']) \)
by (simp add: indRelRTPO.encR)
with \( \text{A1} \) have \( [S'] = T \Longrightarrow \text{SourceTerm} S' \lesssim [\cdot]RT<\text{TRel}> P' \)
by simp
moreover have \( ([S'], T) \in \text{TRel}^+ \Longrightarrow \text{SourceTerm} S' \lesssim [\cdot]RT<\text{TRel}> P' \)
proof
assume \( ([S'], T) \in \text{TRel}^+ \)
\( \text{hence} \ \text{TargetTerm} ([S']) \lesssim [\cdot]RT<\text{TRel}> \text{Target Term} T \)
by (rule transitive-closure-of-TRel-to-indRelRTPO)
with \( \text{A5} \) have \( \text{SourceTerm} S' \lesssim [\cdot]\text{RT}<\text{TRel}> \text{Target Term} T \)
by (rule indRelRTPO.trans)
with \( \text{A1} \) show \( \text{SourceTerm} S' \lesssim [\cdot]\text{RT}<\text{TRel}> P' \)
by simp
qed
ultimately show \( \text{SourceTerm} S' \lesssim [\cdot]\text{RT}<\text{TRel}> P' \)
by blast
qed
hence \( (P', \ \text{SourceTerm} S') \in \text{(indRelRTPO TRel)}^{-1} \)
by simp
ultimately
show \( \exists Q'. \ \text{SourceTerm} S \longrightarrow (\text{STCal Source Target}) \ Q' \land (P', Q') \in \text{(indRelRTPO TRel)}^{-1} \)
by blast
next
case (source \( S) \)
then obtain \( S' \) where \( B1: \ S' \in S \) \( P' \)
by (auto simp add: STCal-step(I))
hence \( (P', P') \in \text{(indRelRTPO TRel)}^{-1} \)
by (simp add: indRelRTPO.source)
with source
show \( \exists Q'. \ \text{SourceTerm} S \longrightarrow (\text{STCal Source Target}) \ Q' \land (P', Q') \in \text{(indRelRTPO TRel)}^{-1} \)
by blast
next
case (target \( T1 T2) \)
assume TargetTerm \( T2 \longrightarrow (\text{STCal Source Target}) \ P' \)
from this obtain \( T2' \) where \( C1: \ T2' \in T \) \( P' \) and \( C2: \ T2 \longrightarrow \text{Target T2'} \)
by (auto simp add: STCal-step(2))
assume \( (T1, T2) \in \text{TRel} \)
hence \( (T2, T1) \in (\text{TRel})^{-1} \)
by simp
with \( C2 \) sim obtain \( T1' \) where \( C3: \ T1 \longrightarrow \text{Target T1'} \) and \( C4: \ (T2', T1') \in (\text{TRel})^{-1} \)
by blast
from \( C3 \) have TargetTerm \( T1 \longrightarrow (\text{STCal Source Target}) \ (\text{TargetTerm} T1') \)
by (simp add: STCal-step(2))
moreover from \( C4 \) have \( (T1', T2') \in \text{TRel}^+ \)
by induct
hence TargetTerm \( T1' \lesssim [\cdot]\text{RT}<\text{TRel}> \text{TargetTerm} T2' \)

by (rule transitive-closure-of-TRel-to-indRelRTPO)
with Cl have \( (P', \text{TargetTerm} \ T1') \in (\text{indRelRTPO} \ TRel)^{-1} \)
by simp
ultimately
show \( \exists \ Q'. \text{TargetTerm} \ T1 \mapsto (\text{STCal} \ Source \ Target) \ Q' \wedge (P', Q') \in (\text{indRelRTPO} \ TRel)^{-1} \)
by blast
next
case \( (\text{trans} \ P \ Q \ R') \)
assume \( R \mapsto (\text{STCal} \ Source \ Target) \ R' \)
and \( \land \, R', \, R' \mapsto (\text{STCal} \ Source \ Target) \ R' \)
\( \implies \exists \ Q'. \, Q \mapsto (\text{STCal} \ Source \ Target) \ Q' \wedge (R', Q') \in (\text{indRelRTPO} \ TRel)^{-1} \)
from this obtain \( Q' \) where \( D1: Q \mapsto (\text{STCal} \ Source \ Target) \ Q' \)
and \( D2: (R', Q') \in (\text{indRelRTPO} \ TRel)^{-1} \)
by blast
assume \( \land \, Q', \, Q \mapsto (\text{STCal} \ Source \ Target) \ Q' \)
\( \implies \exists \ P', \, P \mapsto (\text{STCal} \ Source \ Target) \ P' \wedge (Q', P') \in (\text{indRelRTPO} \ TRel)^{-1} \)
with \( D1 \) obtain \( P' \) where \( D3: P \mapsto (\text{STCal} \ Source \ Target) \ P' \)
and \( D4: (Q', P') \in (\text{indRelRTPO} \ TRel)^{-1} \)
by blast
from \( D4 \, D2 \) have \( (R', P') \in (\text{indRelRTPO} \ TRel)^{-1} \)
by (simp add: indRelRTPO.trans[where \( \text{P} = P' \) and \( Q = Q' \) and \( R = R' \)])
with \( D3 \) show \( \exists \ P'. \, P \mapsto (\text{STCal} \ Source \ Target) \ P' \wedge (R', P') \in (\text{indRelRTPO} \ TRel)^{-1} \)
by blast
qed
qed
next
have \( \forall \, S. \, \text{SourceTerm} \ S \subseteq \{\} \ 	ext{RT} < \text{TRel} \, \text{TargetTerm} \ (\{S\}) \)
by (simp add: indRelRTPO.encR)
moreover have \( \forall \, S \, T. \, \text{SourceTerm} \ S \subseteq \{\} \ 	ext{RT} < \text{TRel} \, \text{TargetTerm} \ T \rightarrow (\{S\}, T) \in \text{TRel}^+ \)
using indRelRTPO-to-TRel(2)[where \( \text{TRel=} \text{TRel} \) trans-closure-of-TRel-refl-cond]
by simp
moreover
assume \( \text{sim: strong-reduction-simulation} \ ((\text{indRelRTPO} \ TRel)^{-1}) \ (\text{STCal} \ Source \ Target) \)
ultimately have \( \text{strongly-operational-sound} \ (\text{TRel}^+) \)
using strong-reduction-simulation-impl-SOSou[where \( \text{Rel=} \text{indRelRTPO} \ TRel \) and \( \text{TRel=} \text{TRel} \)]
by simp
moreover from \( \text{sim} \) have \( \text{strong-reduction-simulation} \ ((\text{TRel}^+)^{-1}) \ \text{Target} \)
using indRelRTPO-impl-TRel-is-strong-reduction-simulation-rev[where \( \text{TRel=} \text{TRel} \)]
by simp
ultimately
show \( \text{strongly-operational-sound} \ (\text{TRel}^+) \wedge \text{strong-reduction-simulation} \ ((\text{TRel}^+)^{-1}) \ \text{Target} \)
by simp
qed

lemma \( \text{(in encoding) SOSou-iff-strong-reduction-simulation:} \)
fixes \( \text{TRel: ('procT} \times 'procT) \) set
shows \( \text{(strongly-operational-sound} \ (\text{TRel}^+) \wedge \text{strong-reduction-simulation} \ ((\text{TRel}^+)^{-1}) \ \text{Target}) \)
\( = (\exists \, \text{Rel}. \ (\forall \, S. \, (\text{SourceTerm} \ S, \text{TargetTerm} \ (\{S\})) \in \text{Rel}) \)
\( \land (\forall \, \text{TargetTerm} \ T1, \text{TargetTerm} \ T2 \in \text{Rel} \rightarrow (\text{TargetTerm} \ T1, \text{TargetTerm} \ T2) \in \text{Rel}) \)
\( \land (\forall \, \text{TargetTerm} \ T1, \text{TargetTerm} \ T2 \in \text{Rel} \rightarrow (\text{TargetTerm} \ T1, \text{TargetTerm} \ T2) \in \text{Rel}^+ \)
\( \land (\forall \, S \, T. \, (\text{SourceTerm} \ S, \text{TargetTerm} \ T) \in \text{Rel} \rightarrow (\{S\}, T) \in \text{TRel}^+) \)
\( \land \text{strong-reduction-simulation} \ (\text{Rel}^{-1}) \ (\text{STCal} \ Source \ Target) \)
proof (rule iffI, erule conjE)
have \( \forall \, S. \, (\text{SourceTerm} \ S, \text{TargetTerm} \ (\{S\})) \in \text{indRelRTPO} \ TRel \)
by (simp add: indRelRTPO.encR)
moreover have \( \forall \, \text{TargetTerm} \ T1, \text{TargetTerm} \ T2 \in \text{TRel} \rightarrow (\text{TargetTerm} \ T1 \subseteq \{\} \ 	ext{RT} < \text{TRel} \, \text{TargetTerm} \ T2 \)
by (simp add: indRelRTPO.target)
moreover have \( \forall \, \text{TargetTerm} \ T1 \subseteq \{\} \ 	ext{RT} < \text{TRel} \rightarrow (\text{TargetTerm} \ T2 \rightarrow (\text{TargetTerm} \ T1, \text{TargetTerm} \ T2) \in \text{TRel}^+) \)
using indRelRTPO-to-TRel(4)[where \( \text{TRel=} \text{TRel} \)]
by simp
moreover have \( \forall \, S \, T. \, (\text{SourceTerm} \ S \subseteq \{\} \ 	ext{RT} < \text{TRel} \, \text{TargetTerm} \ T \rightarrow (\{S\}, T) \in \text{TRel}^+ \)

using \textit{indRelRTPO-to-TRel}(\exists)[\textbf{where} \ TRel=TRel] \ trans-closure-of-TRel-refl-cond
by simp
moreover assume strongly-operational-sound (TRel*)
and strong-reduction-simulation ((TRel*)^{-1}) Target
hence strong-reduction-simulation (((\exists \textit{indRelRTPO} \ TRel)^{-1}) \ STCal \ Source \ Target)
using \textit{SOSou-iff-inverse-of-indRelRTPO-is-strong-reduction-simulation}[\textbf{where} \ TRel=TRel]
by simp
ultimately show \exists \textit{Rel}. (\forall S. \ (SourceTerm \ S, \ TargetTerm \ ([S])) \in \textit{Rel})
\land (\forall T1 T2. \ (T1, \ T2) \in \textit{TRel} \implies (\textit{TargetTerm} \ T1, \ \textit{TargetTerm} \ T2) \in \textit{Rel})
\land (\forall T1 T2. \ (\textit{TargetTerm} \ T1, \ \textit{TargetTerm} \ T2) \in \textit{Rel} \implies (T1, \ T2) \in \textit{TRel}^*)
\land (\forall S \ T. \ (SourceTerm \ S, \ TargetTerm \ T) \in \textit{Rel} \implies ([S], \ T) \in \textit{TRel}^*)
\land \textit{strong-reduction-simulation} (Rel^{-1}) \ (\textit{STCal \ Source \ Target})
by blast
from this obtain \textit{Rel} where A1: \forall S. \ (SourceTerm \ S, \ TargetTerm \ ([S])) \in \textit{Rel}
and A2: \forall T1 T2. \ (T1, \ T2) \in \textit{TRel} \implies (\textit{TargetTerm} \ T1, \ \textit{TargetTerm} \ T2) \in \textit{Rel}
and A3: \forall T1 T2. \ (\textit{TargetTerm} \ T1, \ \textit{TargetTerm} \ T2) \in \textit{Rel} \implies (T1, \ T2) \in \textit{TRel}^*
and A4: \forall S T. \ (SourceTerm \ S, \ TargetTerm \ T) \in \textit{Rel} \implies ([S], \ T) \in \textit{TRel}^*
and A5: \textit{strong-reduction-simulation} (Rel^{-1}) \ (\textit{STCal \ Source \ Target})
by blast
from A1 A4 A5 have strongly-operational-sound (TRel*)
using \textit{strong-reduction-simulation-impl-SOSou}[\textbf{where} \ Rel=\textit{Rel} \ \textbf{and} \ TRel=\textit{TRel}]
by simp
moreover from A2 A3 A5 have strong-reduction-simulation ((TRel*)^{-1}) Target
using \textit{rel-with-target-impl-transC-TRel-is-strong-reduction-simulation-rev}[\textbf{where} \ Rel=\textit{Rel} \ \textbf{and} \ TRel=\textit{TRel}]
by simp
ultimately show strongly-operational-sound (TRel*) \ \land \ \textit{strong-reduction-simulation} ((TRel*)^{-1}) Target
by simp
qed

lemma \textit{(in encoding) SOSou-modulo-TRel-iff-strong-reduction-simulation}:
shows (\exists TRel. \ \textit{strongly-operational-sound} (TRel*)
\land \ \textit{strong-reduction-simulation} ((TRel*)^{-1}) Target
= (\exists \textit{Rel}. \ (\forall S. \ (SourceTerm \ S, \ TargetTerm \ ([S])) \in \textit{Rel})
\land (\forall S \ T. \ (SourceTerm \ S, \ TargetTerm \ T) \in \textit{Rel} \implies (\textit{TargetTerm} \ ([S]), \ \textit{TargetTerm} \ T) \in \textit{Rel}^*)
\land \ \textit{strong-reduction-simulation} (Rel^{-1}) \ (\textit{STCal \ Source \ Target}))
proof (rule iffI)
assume \exists TRel. \ \textit{strongly-operational-sound} (TRel*)
\land \ \textit{strong-reduction-simulation} ((TRel*)^{-1}) Target
from this obtain \textit{Rel} where strongly-operational-sound (TRel*)
\land \ \textit{strong-reduction-simulation} ((TRel*)^{-1}) Target
by blast
hence strong-reduction-simulation (((\exists \textit{indRelRTPO} \ TRel)^{-1}) \ STCal \ Source \ Target)
using \textit{SOSou-iff-inverse-of-indRelRTPO-is-strong-reduction-simulation}[\textbf{where} \ TRel=TRel]
by simp
moreover have \forall S. \ (SourceTerm \ S, \ TargetTerm \ ([S])) \in \textit{indRelRTPO} \ TRel
by (simp add: \textit{indRelRTPO}.encR)
moreover have \forall S \ T. \ (SourceTerm \ S, \ TargetTerm \ T) \in \textit{indRelRTPO} \ TRel
\implies (\textit{TargetTerm} \ ([S]), \ \textit{TargetTerm} \ T) \in (\textit{indRelRTPO} \ TRel)^*
using \textit{indRelRTPO-relates-source-target}[\textbf{where} \ TRel=TRel]
by simp
ultimately show \exists \textit{Rel}. \ (\forall S. \ (SourceTerm \ S, \ TargetTerm \ ([S])) \in \textit{Rel})
\land (\forall S \ T. \ (SourceTerm \ S, \ TargetTerm \ T) \in \textit{Rel}
\implies (\textit{TargetTerm} \ ([S]), \ \textit{TargetTerm} \ T) \in \textit{Rel}^*)

174
\( \land \text{strong-reduction-simulation (Rel}^{-1}) \) (STCal Source Target)

by blast

next

assume \( \exists \rel. (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \rel) \)

\( \land (\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \rel \rightarrow (\text{TargetTerm } ([S]), \text{TargetTerm } T) \in \rel^{-1}) \)

\( \land \text{strong-reduction-simulation (Rel}^{-1}) \) (STCal Source Target)

from this obtain \( \rel \) where \( A1: \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \rel \)

and \( A2: (\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \rel \rightarrow (\text{TargetTerm } ([S]), \text{TargetTerm } T) \in \rel^{-1}) \)

and \( A3: \text{strong-reduction-simulation (Rel}^{-1}) \) (STCal Source Target)

by blast

from \( A2 \) obtain \( TRel \) where \( \forall T1 T2. (T1, T2) \in TRel \rightarrow (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \rel \)

and \( \forall T1 T2. (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \rel \rightarrow (T1, T2) \in \rel^{-1} \)

and \( \forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \rel \rightarrow ([S], T) \in \rel^{-1} \)

using \( \text{target-relation-from-source-target-relation [where Rel}=\text{Rel}] \)

by blast

with \( A1 A3 \)

have \( \text{strongly-operational-sound (TRel}^{*}) \land \text{strong-reduction-simulation ((TRel}^{*})^{-1}) \) Target

using \( \text{SOSou-iff-strong-reduction-simulation [where TRel}=\text{TRel}] \)

by blast

thus \( \exists TRel. \text{strongly-operational-sound (TRel}^{*}) \land \text{strong-reduction-simulation ((TRel}^{*})^{-1}) \) Target

by blast

qed

8.5 Weak Operational Correspondence vs Correspondence Similarity

If there exists a relation that relates at least all source terms and their literal translations, includes TRel, and is a correspondence simulation then the encoding is weakly operational corresponding w.r.t. TRel.

lemma (in encoding) weak-reduction-correspondence-simulation-impl-WOC:

fixes \( \rel :: ((\text{procS} , \text{procT} ) \text{Proc} \times (\text{procS} , \text{procT} ) \text{Proc} ) \text{set} \)

and \( TRel :: (\text{procT} \times \text{procT} ) \text{set} \)

assumes \( \text{enc: } \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \rel \)

and \( \rel: (\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \rel \rightarrow ([S], T) \in \rel^{-1}) \)

and \( cs: \text{weak-reduction-correspondence-simulation Rel (STCal Source Target)} \)

shows \( \text{weakly-operational-corresponding (TRel}^{*}) \)

proof

from \( \text{enc TRel cs show operational-complete (TRel}^{*}) \)

using \( \text{weak-reduction-simulation-impl-OCom [where TRel}=\text{TRel}] \)

by simp

next

show \( \text{weakly-operational-sound (TRel}^{*}) \)

proof clarify

fix \( S T \)

from \( \text{enc have (SourceTerm } S, \text{TargetTerm } ([S])) \in \rel \)

by simp

moreover assume \( [S] \mapsto \text{Target}* T \)

hence \( \text{TargetTerm } ([S]) \mapsto (\text{STCal Source Target})* (\text{TargetTerm } T) \)

by (simp add: STCal-steps(2))

ultimately obtain \( P' Q' \) where \( A1: \text{SourceTerm } S \mapsto (\text{STCal Source Target})* P' \)

and \( A2: \text{TargetTerm } T \mapsto (\text{STCal Source Target})* Q' \) and \( A3: (P', Q') \in \rel \)

using \( cs \)

by blast

from \( A1 \) obtain \( S' \) where \( A4: S' \in S P' \) and \( A5: S \mapsto \text{Source}* S' \)

by (auto simp add: STCal-steps(1))

from \( A2 \) obtain \( T' \) where \( A6: T' \in T Q' \) and \( A7: T \mapsto \text{Target}* T' \)

by (auto simp: STCal-steps(2))

from \( \text{tRel A3 A4 A6 have ([S'], T') \in TRel}^{*} \)

by simp
with A5 A7 show \( \exists S' T'. S \rightarrow^* Source \ast S' \land T \rightarrow^* Target \ast T' \land ([S'], T') \in TRel^+ \)
by blast
qed

An encoding is weakly operational corresponding w.r.t. a correspondence simulation on target terms TRel if there exists a relation, like indRelRTPO, that relates at least all source terms and their literal translations, includes TRel, and is a correspondence simulation.

**Lemma (in encoding)** WOC-iff-indRelRTPO-is-reduction-correspondence-simulation:

- fixes TRel :: ('procT \times 'procT) set
- shows (weakly-operational-corresponding (TRel)) ∧ weak-reduction-correspondence-simulation (TRel^+) Target
  = weak-reduction-correspondence-simulation (indRelRTPO TRel) (STCal Source Target)

**Proof**

- **assumption** woc: weakly-operational-corresponding (TRel)
- **and** csi: weak-reduction-correspondence-simulation (TRel^+) Target
- **show** weak-reduction-correspondence-simulation (indRelRTPO TRel) (STCal Source Target)

**Proof**

- **from** woc csi show sim: weak-reduction-simulation (indRelRTPO TRel) (STCal Source Target)
  using OCom-iff-indRelRTPO-is-weak-reduction-simulation[where TRel=TRel]
  by simp
- **show** \( \forall P Q Q'. P \leq_{[\cdot]} RT < TRel > Q \land Q \rightarrow STCal Source Target \ast Q' \rightarrow (\exists P'' Q''). P \rightarrow STCal Source Target \ast P'' \land Q' \rightarrow STCal Source Target \ast Q'' \land P'' \leq_{[\cdot]} RT < TRel > Q'' \)

**Proof**

- **clarify**
- **fix** P Q Q'
- **assume** P \( \leq_{[\cdot]} RT < TRel > Q \) and Q \( \rightarrow STCal Source Target \ast Q' \)
- **thus** \( \exists P'' Q''. P \rightarrow STCal Source Target \ast P'' \land Q' \rightarrow STCal Source Target \ast Q'' \land P'' \leq_{[\cdot]} RT < TRel > Q'' \)

**Proof**

- **induct arbitrary: Q'**
- **case** (encR S)
- **assume** TargetTerm ([S]) \( \rightarrow STCal Source Target \ast Q' \)
- **from** this obtain T where A1: T \( \in T Q' \) and A2: [S] \( \rightarrow Target \ast T \)
  by (auto simp add: STCal-steps(2))
- **from** A2 woc obtain S' T' where A3: S \( \rightarrow Source \ast S' \) and A4: T \( \rightarrow Target \ast T' \)
  and A5: ([S'], T') \( \in TRel^+ \)
  by blast
- **from** A3 have SourceTerm S \( \rightarrow STCal Source Target \ast (SourceTerm S') \)
  by (simp add: STCal-steps(1))
- **moreover from** A4 have TargetTerm T \( \rightarrow STCal Source Target \ast (TargetTerm T') \)
  by (simp add: STCal-steps(2))
- **moreover have** SourceTerm S' \( \leq_{[\cdot]} RT < TRel > TargetTerm T' \)

**Proof**

- **have** A6: SourceTerm S' \( \leq_{[\cdot]} RT < TRel > TargetTerm ([S']) \)
  by (rule indRelRTPO.encR)
- **from** A5 have [S'] = T' \( \land ([S'], T') \in TRel^+ \)
  using rtrancl-eq-or-trancl[of [S'] T' TRel]
  by blast
- **moreover from** A6 have [S'] = T' \( \Rightarrow SourceTerm S' \leq_{[\cdot]} RT < TRel > TargetTerm T' \)
  by simp
- **moreover have** ([S'], T') \( \in TRel^+ \Rightarrow SourceTerm S' \leq_{[\cdot]} RT < TRel > TargetTerm T' \)

**Proof**

- **assume** ([S'], T') \( \in TRel^+ \)
- **hence** TargetTerm ([S']) \( \leq_{[\cdot]} RT < TRel > TargetTerm T' \)
  by (simp add: transitive-closure-of-TRel-to-indRelRTPO[where TRel=TRel])
- **with** A6 show SourceTerm S' \( \leq_{[\cdot]} RT < TRel > TargetTerm T' \)
  by (rule indRelRTPO.trans)
- **qed**
- **ultimately show** SourceTerm S' \( \leq_{[\cdot]} RT < TRel > TargetTerm T' \)
  by blast
qed
ultimately show \( \exists P'' Q'' . \text{SourceTerm} S \xrightarrow{(\text{STCal Source Target})} P'' \wedge Q' \xrightarrow{(\text{STCal Source Target})} Q'' \wedge P'' \leq_s \cdot [\cdot]RT < \cdot \text{TRel} \cdot \cdot Q'' \)

using \( A1 \)
by blast

next
case (source \( S \))
assume \( B1 : \text{SourceTerm} S \xrightarrow{(\text{STCal Source Target})} Q' \)
moreover have \( Q'' \xrightarrow{(\text{STCal Source Target})} Q'' \)
by (rule steps-refl)
moreover from \( B1 \) obtain \( S' \) where \( S' \in S Q' \)
by (auto simp add: STCal-steps(1))
hence \( Q' \leq_s \cdot [\cdot]RT < \cdot \text{TRel} \cdot \cdot Q'' \)
by (simp add: indRelRTPO.source)
ultimately show \( \exists P'' Q'' . \text{SourceTerm} S \xrightarrow{(\text{STCal Source Target})} P'' \wedge Q' \xrightarrow{(\text{STCal Source Target})} Q'' \wedge P'' \leq_s \cdot [\cdot]RT < \cdot \text{TRel} \cdot \cdot Q'' \)
by blast

next
case (target \( T1 \) \( T2 \))
assume \( \text{TargetTerm} T2 \xrightarrow{(\text{STCal Source Target})} Q' \)
from \( \text{this} \) obtain \( T2' \) where \( C1 : T2' \in T Q' \) and \( C2 : T2 \xrightarrow{} \text{Target} * T2' \)
by (auto simp add: STCal-steps(2))
assume \( (T1, T2) \in \text{TRel} \)
hence \( (T1, T2) \in \text{TRel}^+ \)
by simp
with \( C2 \) csi obtain \( T1' T2'' \) where \( C3 : T1 \xrightarrow{} \text{Target} * T1' \) and \( C4 : T2 \xrightarrow{} \text{Target} * T2'' \)
and \( C5 : (T1', T2'') \in \text{TRel}^+ \)
by blast
from \( C3 \) have \( \text{TargetTerm} T1 \xrightarrow{(\text{STCal Source Target})} (\text{TargetTerm} T1') \)
by (simp add: STCal-steps(2))
moreover from \( C1 C4 \) have \( Q' \xrightarrow{(\text{STCal Source Target})} (\text{TargetTerm} T2'') \)
by (simp add: STCal-steps(2))
m moreover from \( C5 \) have \( \text{TargetTerm} T1' \leq_s \cdot [\cdot]RT < \cdot \text{TRel} \cdot \cdot (\text{TargetTerm} T2'') \)
by (simp add: transitive-closure-of-TRel-to-indRelRTPO)
ultimately show \( \exists P'' Q'' . \text{TargetTerm} T1 \xrightarrow{(\text{STCal Source Target})} P'' \wedge Q' \xrightarrow{(\text{STCal Source Target})} Q'' \wedge P'' \leq_s \cdot [\cdot]RT < \cdot \text{TRel} \cdot \cdot Q'' \)
by blast

next
case (trans \( P \) \( Q \) \( R \) \( R' \))
assume \( R \xrightarrow{} (\text{STCal Source Target}) * R' \)
and \( \wedge R' . R \xrightarrow{} (\text{STCal Source Target}) * R' \Rightarrow \exists Q'' R''. Q \xrightarrow{} (\text{STCal Source Target}) * Q'' \wedge R' \xrightarrow{} (\text{STCal Source Target}) * R'' \wedge Q'' \leq_s \cdot [\cdot]RT < \cdot \text{TRel} \cdot \cdot R'' \)
and \( \wedge Q'. Q \xrightarrow{} (\text{STCal Source Target}) * Q' \Rightarrow \exists P'' Q''. P \xrightarrow{} (\text{STCal Source Target}) * P'' \wedge Q' \xrightarrow{} (\text{STCal Source Target}) * Q'' \wedge P'' \leq_s \cdot [\cdot]RT < \cdot \text{TRel} \cdot \cdot Q'' \)
moreover have \( \text{trans} (\text{indRelRTPO TRel}) \)
using \( \text{indRelRTPO} . \text{trans} \)
unfolding \( \text{trans-def} \)
by blast
ultimately show \( \text{?case} \)
using \( \text{sim reduction-correspondence-simulation-condition-trans\ where} P=P \) and
\( \text{Rel} = \text{indRelRTPO} \) \( \text{TRel} \) and \( \text{Cal} = \text{STCal Source Target} \) and \( Q=Q \) and \( R=R \)
by blast
qed

qed

next
case (source \( S \))
assume \( \text{csi} : \text{weak-reduction-correspondence-simulation} (\text{indRelRTPO TRel}) (\text{STCal Source Target}) \)
show \( \text{weakly-operational-corresponding (TRel^+)} \) \text{Target} 
proof
have \( \forall S . \text{SourceTerm} S \leq_s \cdot [\cdot]RT < \cdot \text{TRel} \cdot \cdot \text{TargetTerm} ([S]) \)
by (simp add: indRelRTPO.enclR)
moreover have \( \forall S T. \text{SourceTerm} S \leq_{\text{\textit{RT}}<\text{TRel}} \text{TargetTerm} T \rightarrow ([S], T) \in \text{TRel}^* \)
  using indRelRTPO-to-TRel(2)[where TRel=TRel] trans-closure-of-TRel-refl-cond
by simp
ultimately show weakly-operational-corrresponding (TRel*)
  using weak-reduction-correspondence-simulation-impl-WOC[where Rel=indRelRTPO TRel and TRel=TRel] csi
by simp
next
from csi show weak-reduction-correspondence-simulation (TRel*) Target
  using indRelRTPO-impl-TRel-is-weak-reduction-correspondence-simulation[where TRel=TRel] by simp
qed

lemma (in encoding) WOC-iff-indRTPO-is-reduction-correspondence-simulation:
  fixes TRel :: \("\langle\text{proc} T \times \text{\textit{proc} T}\rangle\) set
shows (weakly-operational-corrresponding (TRel*))
  \(\wedge\) weak-reduction-correspondence-simulation (TRel*) Target
= (\(\exists\ Rel. \ (\forall S. \ (\text{SourceTerm} S, \text{TargetTerm} ([S]) \in \text{TRel})\)
  \(\wedge\ (\forall T1 T2. \ (T1, T2) \in \text{TRel} \rightarrow (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel})\)
  \(\wedge\ (\forall T1 T2. \ (T1, T2) \in \text{TRel} \rightarrow (T1, T2) \in \text{TRel}^*)\)
  \(\wedge\ (\forall S T. \ (\text{SourceTerm} S, \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}^*)\)
  \(\wedge\) weak-reduction-correspondence-simulation Rel (STCal Source Target))
proof (rule iffI, erule conjE)
  have \(\forall S. \ (\text{SourceTerm} S, \text{TargetTerm} ([S]) \in \text{TRel} \rightarrow \text{TRel})\)
  by (simp add: indRelRTPO enclR)
moreover have \(\forall T1 T2. \ (T1, T2) \in \text{TRel} \rightarrow \text{TargetTerm} T1 \leq_{\text{\textit{RT}}<\text{TRel}} \text{TargetTerm} T2\)
  by (simp add: indRelRTPO target)
moreover have \(\forall T1 T2. \ (\text{TargetTerm} T1 \leq_{\text{\textit{RT}}<\text{TRel}} \text{TargetTerm} T2 \rightarrow (T1, T2) \in \text{TRel}^*)\)
  using indRelRTPO-to-TRel(4)[where TRel=TRel] by simp
moreover have \(\forall S T. \ (\text{SourceTerm} S, \text{TargetTerm} T) \leq_{\text{\textit{RT}}<\text{TRel}} \text{TargetTerm} T \rightarrow ([S], T) \in \text{TRel}^*\)
  using indRelRTPO-to-TRel(2)[where TRel=TRel] trans-closure-of-TRel-refl-cond
  by simp
moreover assume weakly-operational-corrresponding (TRel*)
  and weak-reduction-correspondence-simulation (TRel*) Target
hence weak-reduction-correspondence-simulation (indRelRTPO TRel) (STCal Source Target)
using WOC-iff-indRTPO-is-reduction-correspondence-simulation[where TRel=TRel] by simp
ultimately show \(\exists\ Rel. \ (\forall S. \ (\text{SourceTerm} S, \text{TargetTerm} ([S]) \in \text{Rel})\)
  \(\wedge\ (\forall T1 T2. \ (T1, T2) \in \text{TRel} \rightarrow (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel})\)
  \(\wedge\ (\forall T1 T2. \ (T1, T2) \in \text{TRel} \rightarrow (T1, T2) \in \text{TRel}^*)\)
  \(\wedge\ (\forall S T. \ (\text{SourceTerm} S, \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}^*)\)
  \(\wedge\) weak-reduction-correspondence-simulation Rel (STCal Source Target)
  by blast
next
assume \(\forall S. \ (\text{SourceTerm} S, \text{TargetTerm} ([S]) \in \text{Rel})\)
  \(\wedge\ (\forall T1 T2. \ (T1, T2) \in \text{TRel} \rightarrow (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel})\)
  \(\wedge\ (\forall T1 T2. \ (T1, T2) \in \text{TRel} \rightarrow (T1, T2) \in \text{TRel}^*)\)
  \(\wedge\ (\forall S T. \ (\text{SourceTerm} S, \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}^*)\)
  \(\wedge\) weak-reduction-correspondence-simulation Rel (STCal Source Target)
from this obtain Rel where A1: \(\forall S. \ (\text{SourceTerm} S, \text{TargetTerm} ([S]) \in \text{Rel})\)
  and A2: \(\forall T1 T2. \ (T1, T2) \in \text{TRel} \rightarrow (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel}\)
  and A3: \(\forall T1 T2. \ (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel} \rightarrow (T1, T2) \in \text{TRel}^*\)
  and A4: \(\forall S T. \ (\text{SourceTerm} S, \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}^*\)
  and A5: weak-reduction-correspondence-simulation Rel (STCal Source Target)
  by blast
from A1 A4 A5 have weakly-operational-corrresponding (TRel*)
  using weak-reduction-correspondence-simulation-impl-WOC[where Rel=Rel and TRel=TRel]
  by simp
moreover from A2 A3 A5 have weak-reduction-correspondence-simulation (TRel\+)
Target
  using rel-with-target-impl-transC-TRel-is-weak-reduction-correspondence-simulation
by simp
ultimately show weakly-operational-corresponding (TRel*)
  ∧ weak-reduction-correspondence-simulation (TRel\+) Target
by simp
qed

lemma rel-includes-TRel-modulo-preorder:
  fixes Rel :: ('procS, 'procT) Proc × ('procS, 'procT) Proc set
and TRel :: ('procT × 'procT) set
assumes transT: trans TRel
shows \((\forall T1 T2. (T1, T2) \in TRel \rightarrow (T1, T2) \in \text{Rel}) \\
\land (\forall T1 T2. (T1, T2) \in TRel \rightarrow ((T1, T2) \in TRel\+))\)
  = (TRel = \{(T1, T2). (TargetTerm T1, TargetTerm T2) \in \text{Rel}\})
proof (ruleiffI, erule conjE)
  assume \(\forall T1 T2. (T1, T2) \in TRel \rightarrow (TargetTerm T1, TargetTerm T2) \in \text{Rel}\)
  and \(\forall T1 T2. (TargetTerm T1, TargetTerm T2) \in \text{Rel} \rightarrow (T1, T2) \in TRel\+)
  with transT show TRel = \{(T1, T2). (TargetTerm T1, TargetTerm T2) \in \text{Rel}\}
  using trancl-id[of TRel]
  by blast
next
  assume A: TRel = \{(T1, T2). (TargetTerm T1, TargetTerm T2) \in \text{Rel}\}
  hence \(\forall T1 T2. (T1, T2) \in TRel \rightarrow (T1, T2) \in TRel\+)
  by simp
  moreover from transT A
  have \(\forall T1 T2. (TargetTerm T1, TargetTerm T2) \in \text{Rel} \rightarrow (T1, T2) \in TRel\+)
  using trancl-id[of TRel]
  by blast
  ultimately show \(\forall T1 T2. (T1, T2) \in TRel \rightarrow (TargetTerm T1, TargetTerm T2) \in \text{Rel}\)
  \land (\forall T1 T2. (TargetTerm T1, TargetTerm T2) \in \text{Rel} \rightarrow (T1, T2) \in TRel\+)
  by simp
qed

lemma (in encoding) WOC-wrt-preorder-iff-reduction-correspondence-simulation:
  fixes TRel :: ('procT × 'procT) set
  shows (weakly-operational-corresponding TRel \land preorder TRel
  \land weak-reduction-correspondence-simulation TRel Target)
  = (\exists \text{Rel}. (\forall S T. (\text{SourceTerm S, TargetTerm T} \in \text{Rel} \\
  \land TRel = \{(T1, T2). (TargetTerm T1, TargetTerm T2) \in \text{Rel}\} \\
  \land (\forall S T. (\text{SourceTerm S, TargetTerm T} \in \text{Rel} \rightarrow ([S], T) \in TRel)}
  \land preorder Rel
  \land weak-reduction-correspondence-simulation Rel (\text{STCal Source Target})))
proof (ruleiffI, erule conjE, erule conjE, erule conjE)
  assume A1: operational-complete TRel and A2: weakly-operational-sound TRel
  and A3: preorder TRel and A4: weak-reduction-correspondence-simulation TRel Target
  from A3 have A5: TRel\+ = TRel
  using trancl-id[of TRel]
  unfolding preorder-on-def
  by blast
  with A3 have TRel\+ = TRel
  using trancl-id[of TRel] reflcl-trancl[of TRel]
  unfolding preorder-on-def refl-on-def
  by auto
  with A1 A2 have weakly-operational-corresponding (TRel\+)
  by simp
  moreover from A4 A5 have weak-reduction-correspondence-simulation (TRel\+) Target
  by simp
  ultimately
  have weak-reduction-correspondence-simulation (\text{indRelRTPO TRel} (\text{STCal Source Target})
  using WOC-iff-indRelRTPO-is-reduction-correspondence-simulation[where TRel=TRel]

by blast

moreover have \( \forall S . \) \((\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{indRelRTPO } T\text{Rel}\)
by (simp add: \text{indRelRTPO}.encR)

moreover

have \( T\text{Rel} = \{ (T1, T2), (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{indRelRTPO } T\text{Rel}\}\)

proof auto

fix \( TP \) \( TQ \)

assume \( (TP, TQ) \in T\text{Rel} \)

thus \( \text{TargetTerm } TP \preceq_{[\([\_]\)]} RT < T\text{Rel} > \text{TargetTerm } TQ \)
by (rule \text{indRelRTPO}.target)

next

fix \( TP \) \( TQ \)

assume \( \text{TargetTerm } TP \preceq_{[\([\_]\)]} RT < T\text{Rel} > \text{TargetTerm } TQ \)

with \( A3 \) show \( (TP, TQ) \in T\text{Rel} \)
using \text{indRelRTPO-to-TRel}(4)[where \( T\text{Rel} = T\text{Rel} \) \( \text{trancl-id}[of } T\text{Rel} \)]

unfolding \text{preorder-on-def}
by blast

qed

moreover from \( A3 \)

have \( \forall S T . \) \((\text{SourceTerm } S, \text{TargetTerm } T) \in \text{indRelRTPO } T\text{Rel} \rightarrow ([S], T) \in T\text{Rel}^+ \)
using \text{indRelRTPO-to-TRel}(2)[where \( T\text{Rel} = T\text{Rel} \) \( \text{reflcl-trancl}[of } T\text{Rel} \)]

trans-closure-of-TRel-refl-cond[where \( T\text{Rel} = T\text{Rel} \)]

unfolding \text{preorder-on-def refl-on-def}
by blast

with \( A3 \) have \( \forall S T . \) \((\text{SourceTerm } S, \text{TargetTerm } T) \in \text{indRelRTPO } T\text{Rel} \rightarrow ([S], T) \in T\text{Rel} \)
using \text{trancl-id}[of } T\text{Rel}]

unfolding \text{preorder-on-def}
by blast

moreover from \( A3 \) have \( \text{refl } (\text{indRelRTPO } T\text{Rel}) \)
using \text{indRelRTPO-refl}[of } T\text{Rel}]

unfolding \text{preorder-on-def}
by simp

moreover have \( \text{trans } (\text{indRelRTPO } T\text{Rel}) \)
using \text{indRelRTPO}.trans

unfolding \text{trans-def}
by simp

ultimately show \( \exists \text{Rel} . \) \((\forall S . \) \((\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \)
\land \( \{ (T1, T2) . \) \((\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel} \) \)
\land \( \forall S T . \) \((\text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in T\text{Rel} \)
\land \( \text{preorder } \text{Rel} \)
\land \( \text{weak-reduction-correspondence-simulation } \text{Rel } (\text{STCal Source Target}) \)

unfolding \text{preorder-on-def}
by blast

next

assume \( \exists \text{Rel} . \) \((\forall S . \) \((\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \)
\land \( \{ (T1, T2) . \) \((\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel} \) \)
\land \( \forall S T . \) \((\text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in T\text{Rel} \)
\land \( \text{preorder } \text{Rel} \)
\land \( \text{weak-reduction-correspondence-simulation } \text{Rel } (\text{STCal Source Target}) \)

from \text{this obtain} \text{Rel } \text{where } B1: \( \forall S . \) \((\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \)
and \( B2 : T\text{Rel} = \{ (T1, T2), (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel} \} \)
and \( B3: \forall S T . \) \((\text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in T\text{Rel} \) \and \( B4 : \text{preorder } \text{Rel} \)
and \( B5: \text{weak-reduction-correspondence-simulation } \text{Rel } (\text{STCal Source Target}) \)
by blast

from \( B2 \) \( B4 \) have \( B6 : \text{refl } T\text{Rel} \)

unfolding \text{preorder-on-def refl-on-def}
by blast

from \( B2 \) \( B4 \) have \( B7 : \text{trans } T\text{Rel} \)

unfolding \text{trans-def preorder-on-def}
by blast

hence \( B8 : T\text{Rel}^+ = T\text{Rel} \)
using `trancl-id[of TRel]
by simp
with B6 have `TRel = TRel
using `reflcl-trancl[of TRel]
unfolding refl-on-def
by blast
with B1 B3 B5 have weakly-operational-corresponding TRel
using `weak-reduction-correspondence-simulation-impl-WOC[where Rel=Rel and TRel=TRel]
by simp
moreover from B6 B7 have preorder TRel
unfolding preorder-on-def
by blast
moreover from B2 B5 B7 B8 have weak-reduction-correspondence-simulation TRel Target
using rel-includes-TRel-modulo-preorder[where Rel=Rel and TRel=TRel]
rel-with-target-impl-transC-TRel-is-weak-reduction-correspondence-simulation[where Rel=Rel and TRel=TRel]
by fast
ultimately show weakly-operational-corresponding TRel \land preorder TRel
\land weak-reduction-correspondence-simulation TRel Target
by blast
qed

8.6 (Strong) Operational Correspondence vs (Strong) Bisimilarity

An encoding is operational corresponding w.r.t a weak bisimulation on target terms TRel iff there exists a relation, like indRelRTPO, that relates at least all source terms and their literal translations, includes TRel, and is a weak bisimulation. Thus this variant of operational correspondence ensures that source terms and their translations are weak bisimilar.

lemma (in encoding) OC-iff-indRelRTPO-is-weak-reduction-bisimulation:
  fixes TRel :: (′procT × ′procT) set
  shows (operational-corresponding (TRel*) \land weak-reduction-bisimulation (TRel*) Target)
  = weak-reduction-bisimulation (indRelRTPO TRel) (STCal Source Target)
proof (rule iffI, erule conjE)
  assume ocorr: operational-corresponding (TRel*)
  and bisim: weak-reduction-bisimulation (TRel*) Target
  hence weak-reduction-simulation (indRelRTPO TRel) (STCal Source Target)
  using OCom-iff-indRelRTPO-is-weak-reduction-simulation[where TRel=TRel]
  by simp
  moreover from bisim have weak-reduction-simulation ((TRel*)\(^{-1}\)) Target
  using weak-reduction-bisimulations-impl-inverse-is-simulation[where Rel=TRel]
  by simp
  with ocorr have weak-reduction-simulation ((indRelRTPO TRel)\(^{-1}\)) (STCal Source Target)
  using OSou-iff-inverse-of-indRelRTPO-is-weak-reduction-simulation[where TRel=TRel]
  by simp
  ultimately show weak-reduction-bisimulation (indRelRTPO TRel) (STCal Source Target)
  using weak-reduction-simulations-impl-simulation[where Rel=indRelRTPO TRel]
  by simp
next
  assume bisim: weak-reduction-bisimulation (indRelRTPO TRel) (STCal Source Target)
  hence operational-complete (TRel*) \land weak-reduction-simulation (TRel*) Target
  using OCom-iff-indRelRTPO-is-weak-reduction-simulation[where TRel=TRel]
  by simp
  moreover from bisim
  have weak-reduction-simulation ((indRelRTPO TRel)\(^{-1}\)) (STCal Source Target)
  using weak-reduction-bisimulations-impl-inverse-is-simulation[where Rel=indRelRTPO TRel]
  by simp
  hence operational-sound (TRel*) \land weak-reduction-simulation ((TRel*)\(^{-1}\)) Target
  using OSou-iff-inverse-of-indRelRTPO-is-weak-reduction-simulation[where TRel=TRel]
  by simp
ultimately show operational-corresponding \((\mathcal{T} Rel^+)\) \(\land\) weak-reduction-bisimulation \((\mathcal{T} Rel^+)\) Target
using weak-reduction-simulations-impl-bisimulation[where \(\mathcal{R} el=\mathcal{T} Rel^+)\]
by simp
qed

lemma (in encoding) OC-iff-weak-reduction-bisimulation:
fixes \(\mathcal{T} Rel::('a proc \times 'a proc) set\)
shows \((\text{operational-corresponding }(\mathcal{T} Rel^+)\) \(\land\) weak-reduction-bisimulation \((\mathcal{T} Rel^+)\) Target\)
\(\equiv\) \((\exists \mathcal{R} el. \forall S. \text{(SourceTerm } S, \text{TargetTerm } ([S]) ) \in \mathcal{R} el\)
\(\land\) \((\forall T T_1 T_2. \text{(T1}, T_2) \in \mathcal{T} \mathcal{R} el\rightarrow (\text{TargetTerm } T_1, \text{TargetTerm } T_2) \in \mathcal{R} el\)
\(\land\) \((\forall T T_1 T_2. \text{(TargetTerm } T_1, \text{TargetTerm } T_2) \in \mathcal{R} el\rightarrow (T_1, T_2) \in \mathcal{T} Rel^+)\)
\(\land\) \((\forall S T. \text{(SourceTerm } S, \text{TargetTerm } T) \in \mathcal{R} el\rightarrow ([S], T) \in \mathcal{T} Rel^+)\)
\(\land\) weak-reduction-bisimulation \(\mathcal{R} el\) (\text{STCal Source Target})

proof (rule iffI,erule conjE)
have \(\forall S. \text{(SourceTerm } S, \text{TargetTerm } ([S]) ) \in \text{indRelRTPO } \mathcal{T} \mathcal{R} el\)
by (simp add: indRelRTPO.encR)
moreover have \(\forall T T_1 T_2. \text{(T1}, T_2) \in \mathcal{T} \mathcal{R} el\rightarrow \text{TargetTerm } T_1 \leq[\[\mathcal{R} el<\mathcal{T} Rel> \text{TargetTerm } T_2\]
by (simp add: indRelRTPO.target)
moreover have \(\forall T T_1 T_2. \text{(TargetTerm } T_1, \text{TargetTerm } T_2) \in \mathcal{R} el\rightarrow (T_1, T_2) \in \mathcal{T} Rel^+)\)
using indRelRTPO-to-TRel(4)[where \(\mathcal{R} el=\mathcal{T} Rel\)]
by simp
moreover have \(\forall S T. \text{SourceTerm } S \leq[\[\mathcal{R} el<\mathcal{T} Rel> \text{TargetTerm } T \rightarrow ([S], T) \in \mathcal{T} Rel^+)\]
using indRelRTPO-to-TRel(2)[where \(\mathcal{R} el=\mathcal{T} Rel\)] trans-closure-of-TRel-refl-cond
by simp
moreover assume \text{operational-corresponding } \((\mathcal{T} Rel^+)\)
and \text{weak-reduction-bisimulation } \((\mathcal{T} Rel^+)\) Target
hence \text{weak-reduction-bisimulation } \text{(indRelRTPO } \mathcal{T} \mathcal{R} el\) \((\text{STCal Source Target})\)
using OC-iff-indRelRTPO-is-weak-reduction-bisimulation[where \(\mathcal{R} el=\mathcal{T} Rel\)]
by simp
ultimately show \(\exists \mathcal{R} el. \forall S. \text{(SourceTerm } S, \text{TargetTerm } ([S]) ) \in \mathcal{R} el\)
\(\land\) \((\forall T T_1 T_2. \text{(T1}, T_2) \in \mathcal{T} \mathcal{R} el\rightarrow (\text{TargetTerm } T_1, \text{TargetTerm } T_2) \in \mathcal{R} el\)
\(\land\) \((\forall T T_1 T_2. \text{(TargetTerm } T_1, \text{TargetTerm } T_2) \in \mathcal{R} el\rightarrow (T_1, T_2) \in \mathcal{T} Rel^+)\)
\(\land\) \((\forall S T. \text{(SourceTerm } S, \text{TargetTerm } T) \in \mathcal{R} el\rightarrow ([S], T) \in \mathcal{T} Rel^+)\)
\(\land\) weak-reduction-bisimulation \(\mathcal{R} el\) (\text{STCal Source Target})

next
assume \(\exists \mathcal{R} el. \forall S. \text{(SourceTerm } S, \text{TargetTerm } ([S]) ) \in \mathcal{R} el\)
\(\land\) \((\forall T T_1 T_2. \text{(T1}, T_2) \in \mathcal{T} \mathcal{R} el\rightarrow (\text{TargetTerm } T_1, \text{TargetTerm } T_2) \in \mathcal{R} el\)
\(\land\) \((\forall T T_1 T_2. \text{(TargetTerm } T_1, \text{TargetTerm } T_2) \in \mathcal{R} el\rightarrow (T_1, T_2) \in \mathcal{T} Rel^+)\)
\(\land\) \((\forall S T. \text{(SourceTerm } S, \text{TargetTerm } T) \in \mathcal{R} el\rightarrow ([S], T) \in \mathcal{T} Rel^+)\)
\(\land\) weak-reduction-bisimulation \(\mathcal{R} el\) (\text{STCal Source Target})
from \text{this obtain } \mathcal{R} el \text{where A1: } \forall S. \text{(SourceTerm } S, \text{TargetTerm } ([S]) ) \in \mathcal{R} el\)
and A2: \(\forall T T_1 T_2. \text{(T1}, T_2) \in \mathcal{T} \mathcal{R} el\rightarrow (\text{TargetTerm } T_1, \text{TargetTerm } T_2) \in \mathcal{R} el\)
and A3: \(\forall T T_1 T_2. \text{(TargetTerm } T_1, \text{TargetTerm } T_2) \in \mathcal{R} el\rightarrow (T_1, T_2) \in \mathcal{T} Rel^+)\)
and A4: \(\forall S T. \text{(SourceTerm } S, \text{TargetTerm } T) \in \mathcal{R} el\rightarrow ([S], T) \in \mathcal{T} Rel^+)\)
and A5: \text{weak-reduction-bisimulation } \mathcal{R} el \text{ (STCal Source Target)}
by blast
hence \text{operational-complete } \((\mathcal{T} Rel^+)\)
\(\land\) weak-reduction-simulation \((\mathcal{T} Rel^+)\) Target
using OCom-iff-weak-reduction-simulation[where \(\mathcal{R} el=\mathcal{T} Rel\)]
by blast
moreover from A5 have \text{weak-reduction-simulation } \((\mathcal{T} Rel^{-1})\) \((\text{STCal Source Target})\)
using weak-reduction-bisimulations-impl-inverse-is-simulation[where \(\mathcal{R} el=\mathcal{T} Rel\)]
by simp
with A1 A2 A3 A4 have \text{operational-sound } \((\mathcal{T} Rel^+)\)
\(\land\) weak-reduction-simulation \((\mathcal{T} Rel^+)\) Target
using OSou-iff-weak-reduction-simulation[where \(\mathcal{R} el=\mathcal{T} Rel\)]
by blast
ultimately show operational-corresponding \((\mathcal{T} Rel^+)\)
\(\land\) weak-reduction-bisimulation \((\mathcal{T} Rel^+)\) Target
using weak-reduction-simulations-impl-bisimulation[where \(\mathcal{R} el=\mathcal{T} Rel^+)\]
by simp

qed

lemma (in encoding) OC-wrt-preorder-iff-weak-reduction-bisimulation:
  fixes TRel :: ('procT × 'procT) set
  shows (operational-corresponding TRel ∧ preorder TRel
    ∧ weak-reduction-bisimulation TRel Target)
    = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
      ∧ TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
      ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel)
      ∧ preorder Rel
      ∧ weak-reduction-bisimulation Rel (STCal Source Target))
proof (rule iffI, erule conjE, erule conjE, erule conjE)
  assume A1: operational-complete TRel and A2: operational-sound TRel
  and A3: preorder TRel and A4: weak-reduction-bisimulation TRel Target
  from A3 have A5: TRel⁺ = TRel
    using trancl-id[of TRel]
    unfolding preorder-on-def
    by blast
  with A3 have TRel⁺ = TRel
    using refl-trancl[of TRel]
    unfolding preorder-on-def refl-on-def
    by blast
  with A1 A2 have operational-corresponding (TRel⁺)
    by simp
  moreover from A4 A5 have weak-reduction-bisimulation (TRel⁺) Target
    by simp
  ultimately
  have weak-reduction-bisimulation (indRelRTPO TRel) (STCal Source Target)
    using OC-iff-indRelRTPO-is-weak-reduction-bisimulation[where TRel=TRel]
    by blast
  moreover have ∀ S. SourceTerm S ≤_R<TRel> TargetTerm ([S])
    by (simp add: indRelRTPO.encR)
  moreover
  have TRel = {(T1, T2). TargetTerm T1 ≤_R<TRel> TargetTerm T2}
  proof
    fix TP TQ
    assume (TP, TQ) ∈ TRel
    thus TargetTerm TP ≤_R<TRel> TargetTerm TQ
      by (rule indRelRTPO.target)
  next
    fix TP TQ
    assume TargetTerm TP ≤_R<TRel> TargetTerm TQ
    with A3 show (TP, TQ) ∈ TRel
      using indRelRTPO-to-TRel(4)[where TRel=TRel] trancl-id[of TRel]
      unfolding preorder-on-def
    by blast
  qed
  moreover from A3
  have ∀ S T. SourceTerm S ≤_R<TRel> TargetTerm T → ([S], T) ∈ TRel⁺
    using indRelRTPO-to-TRel(2)[where TRel=TRel] refl-trancl[of TRel]
    trans-closure-of-TRel-refl-cond[where TRel=TRel]
    unfolding preorder-on-def refl-on-def
    by auto
  with A3 have ∀ S T. SourceTerm S ≤_R<TRel> TargetTerm T → ([S], T) ∈ TRel
    using trancl-id[of TRel]
    unfolding preorder-on-def
    by blast
  moreover from A3 have refl (indRelRTPO TRel)
    unfolding preorder-on-def
    by (simp add: indRelRTPO-refl)

183
moreover have \( \text{trans (indRelRTPO TRel)} \)
  using \( \text{indRelRTPO.trans} \)
unfolding \( \text{trans-def} \)
by blast
ultimately show \( \exists \text{Rel.} \ (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}) \land T\text{Rel} = \{(T1, T2). (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\} \land (\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in T\text{Rel}) \land \text{preorder Rel} \land \text{weak-reduction-bisimulation Rel (STCal Source Target)} \)
unfolding \( \text{preorder-on-def} \)
by blast
next
assume \( \exists \text{Rel.} \ (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}) \land T\text{Rel} = \{(T1, T2). (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\} \land (\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in T\text{Rel}) \land \text{preorder Rel} \land \text{weak-reduction-bisimulation Rel (STCal Source Target)} \)
from this obtain \( \text{Rel where } B1: \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \)
and \( B2: T\text{Rel} = \{(T1, T2). (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\} \land (\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in T\text{Rel} \land B4: \text{preorder Rel} \land B5: \text{weak-reduction-bisimulation Rel (STCal Source Target)} \)
by blast
from \( B2 B4 \) have \( B6: \text{refl } T\text{Rel} \)
unfolding \( \text{preorder-on-def refl-on-def} \)
by blast
from \( B2 B4 \) have \( B7: \text{trans } T\text{Rel} \)
unfolding \( \text{trans-def preorder-on-def} \)
by blast
hence \( B8: T\text{Rel}^+ = T\text{Rel} \)
using \( \text{trancl-id[of TRel]} \)
by simp
with \( B6 \) have \( B9: T\text{Rel}^+ = T\text{Rel} \)
using \( \text{refl-trancl[of TRel]} \)
unfolding \( \text{refl-on-def} \)
by blast
with \( B3 \) have \( \forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in T\text{Rel}^+ \)
by simp
moreover from \( B2 B8 \) have \( \forall T1 T2. (T1, T2) \in T\text{Rel} \rightarrow (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel} \land \forall T1 T2. (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel} \rightarrow (T1, T2) \in T\text{Rel}^+ \)
by auto
ultimately have \( \exists \text{Rel.} \ (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}) \land (\forall T1 T2. (T1, T2) \in T\text{Rel} \rightarrow (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}) \land (\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in T\text{Rel}^+ \land \text{weak-reduction-bisimulation Rel (STCal Source Target)} \)
using \( B1 B5 \)
by blast
hence \( \text{operational-corresponding (TRel^+)} \land \text{weak-reduction-bisimulation (TRel^+)} \) \( \text{Target} \)
using \( \text{OC-off-weak-reduction-bisimulation[where TRel=TRel]} \)
by simp
with \( B8 B9 \) have \( \text{operational-corresponding TRel} \land \text{weak-reduction-bisimulation TRel Target} \)
by simp
moreover from \( B6 B7 \) have \( \text{preorder TRel} \)
unfolding \( \text{preorder-on-def} \)
by blast
ultimately show \( \text{operational-corresponding TRel} \land \text{preorder TRel} \land \text{weak-reduction-bisimulation TRel Target} \)
by blast
qed
lemma (in encoding) OC-wrt-equivalence-iff-indRelTEQ-weak-reduction-bisimulation:
  fixes TRel :: '(procT × 'procT) set
  assumes eqT: equivalence TRel
  shows (operational-corresponding TRel ∧ weak-reduction-bisimulation TRel Target) \iff weak-reduction-bisimulation (indRelTEQ TRel) (STCal Source Target)
proof (rule iffI, erule conjE)
  assume oc: operational-corresponding TRel and bisimT: weak-reduction-bisimulation TRel Target
  show weak-reduction-bisimulation (indRelTEQ TRel) (STCal Source Target)
  proof auto
    fix P Q P'
    assume P  \sim [\cdot] T <\ TRel > Q and P \mapsto (STCal Source Target)* P'
    thus \exists Q'. Q \mapsto (STCal Source Target)* Q' ∧ P' \sim [\cdot] T <\ TRel > Q'
    proof (induct arbitrary: P)
      case (encL S)
      assume TTerm S \mapsto (STCal Source Target)* P'
      from this obtain S' where A1: S \mapsto Source* S' and A2: S' \in S P'
      by (auto simp add: STCal-steps(1))
      from A1 oc obtain T where A3: [\cdot] S \mapsto Target* T and A4: ([\cdot] S', T) \in TRel
      by blast
      from A3 have TTarget ([\cdot] S) \mapsto (STCal Source Target)* (TargetTerm T)
      by (simp add: STCal-steps(2))
      moreover have P' \sim [\cdot] T <\ TRel > TargetTerm T
      proof
        from A2 have P' \sim [\cdot] T <\ TRel > TargetTerm ([\cdot] S')
        by (simp add: indRelTEQ.encR)
        moreover from A4 have TargetTerm ([\cdot] S') \sim [\cdot] T <\ TRel > TargetTerm T
        by (rule indRelTEQ.target)
        ultimately show P' \sim [\cdot] T <\ TRel > TargetTerm T
        by (rule indRelTEQ.trans)
      qed
      ultimately show \exists Q'. TargetTerm ([\cdot] S) \mapsto (STCal Source Target)* Q' ∧ P' \sim [\cdot] T <\ TRel > Q'
      by blast
    next
    case (encR S)
    assume TTerm ([\cdot] S) \mapsto (STCal Source Target)* P'
    from this obtain T where B1: [\cdot] S \mapsto Target* T and B2: T \in T P'
    by (auto simp add: STCal-steps(2))
    from B1 oc obtain S' where B3: S \mapsto Source* S' and B4: ([\cdot] S', T) \in TRel
    by blast
    from B3 have SourceTerm S \mapsto (STCal Source Target)* (SourceTerm S')
    by (simp add: STCal-steps(1))
    moreover have P' \sim [\cdot] T <\ TRel > SourceTerm S'
    proof
      from B4 eqT have (T, [\cdot] S') \in TRel
      unfolding equiv-def sym-def
      by blast
      with B2 have P' \sim [\cdot] T <\ TRel > TargetTerm ([\cdot] S')
      by (simp add: indRelTEQ.target)
      moreover have TargetTerm ([\cdot] S') \sim [\cdot] T <\ TRel > SourceTerm S'
      by (rule indRelTEQ.encL)
      ultimately show P' \sim [\cdot] T <\ TRel > SourceTerm S'
      by (rule indRelTEQ.trans)
    qed
    ultimately show \exists Q'. SourceTerm S \mapsto (STCal Source Target)* Q' ∧ P' \sim [\cdot] T <\ TRel > Q'
    by blast
    next
    case (target T1 T2)
    assume TTerm T1 \mapsto (STCal Source Target)* P'
    from this obtain T1' where C1: T1 \mapsto Target* T1' and C2: T1' \in T P'
    by (auto simp add: STCal-steps(2))
    assume (T1, T2) \in TRel
with $C_1 \text{bisim}_T$ obtain $T_2'$ where $C_3: T_2 \rightsquivalence Target \ast T_2'$ and $C_4: (T_1', T_2') \in TRel$

by blast

from $C_3$ have TargetTerm $T_2 \rightsquivalence (STCal Source Target) \ast (TargetTerm T_2')$

by (simp add: STCal-steps(2))

moreover from $C_2$ $C_4$ have $P' \sim[\cdot]T' \text{<} TRel \text{>} TargetTerm T_2'$

by (simp add: indRelTEQ.target)

ultimately show $\exists Q'$. TargetTerm $T_2 \rightsquivalence (STCal Source Target) \ast Q' \land P' \sim[\cdot]T' \text{<} TRel \text{>} Q'$

by blast

next

case (trans $P \ Q \ R$

assume $P \rightsquivalence (STCal Source Target) \ast P' \land \bigwedge Q' \ast (STCal Source Target) \ast Q' \implies \exists Q'. Q \rightsquivalence (STCal Source Target) \ast Q' \land P' \sim[\cdot]T' \text{<} TRel \text{>} Q'$

from this obtain $Q'$ where $D_1: Q \rightsquivalence (STCal Source Target) \ast Q' \land D_2: P' \sim[\cdot]T' \text{<} TRel \text{>} Q'$

by blast

assume $\bigwedge Q'. Q \rightsquivalence (STCal Source Target) \ast Q'$

implies $\exists R'. R \rightsquivalence (STCal Source Target) \ast R' \land Q' \sim[\cdot]T' \text{<} TRel \text{>} R'$

with $D_1$ obtain $R'$ where $D_3: R \rightsquivalence (STCal Source Target) \ast R' \land D_4: Q' \sim[\cdot]T' \text{<} TRel \text{>} R'$

by blast

from $D_2$ $D_4$ have $P' \sim[\cdot]T' \text{<} TRel \text{>} R'$

by (rule indRelTEQ.trans)

with $D_3$ show $\exists R'. R \rightsquivalence (STCal Source Target) \ast R' \land P' \sim[\cdot]T' \text{<} TRel \text{>} R'$

by blast

qed

next

fix $P \ Q \ Q'$

assume $P \sim[\cdot]T' \text{<} TRel \text{>} Q \land Q \rightsquivalence (STCal Source Target) \ast Q'$

thus $\exists P'. P \rightsquivalence (STCal Source Target) \ast P' \land P' \sim[\cdot]T' \text{<} TRel \text{>} Q'$

proof (induct arbitrary: $Q'$)

case (encL $S$

assume TargetTerm $([S]) \rightsquivalence (STCal Source Target) \ast Q'$

from this obtain $T$ where $E_1: [S] \rightsquivalence Target \ast T \text{ and } E_2: T \in T Q'$

by (auto simp add: STCal-steps(2))

from $E_1$ oc obtain $S'$ where $E_3: S \rightsquivalence Source \ast S'$ and $E_4: ([S'], T) \in TRel$

by blast

from $E_3$ have SourceTerm $S \rightsquivalence (STCal Source Target) \ast (SourceTerm S')$

by (simp add: STCal-steps(1))

moreover have SourceTerm $S' \sim[\cdot]T' \text{<} TRel \text{>} Q'$

proof -

have SourceTerm $S' \sim[\cdot]T' \text{<} TRel \text{>} TargetTerm ([S'])$

by (rule indRelTEQ.encR)

moreover from $E_2$ $E_4$ have TargetTerm $([S']) \sim[\cdot]T' \text{<} TRel \text{>} Q'$

by (simp add: indRelTEQ.target)

ultimately show SourceTerm $S' \sim[\cdot]T' \text{<} TRel \text{>} Q'$

by (rule indRelTEQ.trans)

qed

ultimately show $\exists P'$. SourceTerm $S \rightsquivalence (STCal Source Target) \ast P' \land P' \sim[\cdot]T' \text{<} TRel \text{>} Q'$

by blast

next

case (encL $S$

assume SourceTerm $S \rightsquivalence (STCal Source Target) \ast Q'$

from this obtain $S'$ where $F_1: S \rightsquivalence Source \ast S'$ and $F_2: S' \in S Q'$

by (auto simp add: STCal-steps(1))

from $F_1$ oc obtain $T$ where $F_3: [S] \rightsquivalence Target \ast T \text{ and } F_4: ([S'], T) \in TRel$

by blast

from $F_3$ have TargetTerm $([S]) \rightsquivalence (STCal Source Target) \ast (TargetTerm T)$

by (simp add: STCal-steps(2))

moreover have TargetTerm $T \sim[\cdot]T' \text{<} TRel \text{>} Q'$

proof -

from $F_4$ eqT have $(T, [S']) \in TRel$

unfolding equiv-def sym-def
by blast

hence TargetTerm T \sim_{\cdot \cdot} T < \text{TRel} > TargetTerm ([S])

by (rule indRelTEQ.target)

moreover from F2 have TargetTerm ([S]) \sim_{\cdot \cdot} T < \text{TRel} > Q'

by (simp add: indRelTEQ.encR)

ultimately show TargetTerm T \sim_{\cdot \cdot} T < \text{TRel} > Q'

by (rule indRelTEQ.trans)

qed

ultimately show \exists P'. TargetTerm ([S]) \mapsto (\text{STCal Source Target}) \ast P' \land P' \sim_{\cdot \cdot} T < \text{TRel} > Q'

by blast

next

case (target T1 T2)

assume TargetTerm T2 \mapsto (\text{STCal Source Target}) \ast Q'

from this obtain T2' where G1: T2 \mapsto Target \ast T2' and G2: T2' \in T Q'

by (auto simp add: STCal-steps(2))

assume (T1, T2) \in \text{TRel}

with G3 \text{ bisimT} obtain T1' where G3: T1 \mapsto Target \ast T1' and G4: (T1', T2') \in \text{TRel}

by blast

from G3 have TargetTerm T1 \mapsto (\text{STCal Source Target}) \ast (TargetTerm T1')

by (simp add: STCal-steps(2))

moreover from G2 G4 have TargetTerm T1' \sim_{\cdot \cdot} T < \text{TRel} > Q'

by (simp add: indRelTEQ.target)

ultimately show \exists P'. TargetTerm T1 \mapsto (\text{STCal Source Target}) \ast P' \land P' \sim_{\cdot \cdot} T < \text{TRel} > Q'

by blast

next

case (trans P Q R R')

assume R \mapsto (\text{STCal Source Target}) \ast R'

and \( \forall R'. R \mapsto (\text{STCal Source Target}) \ast R' \)

\( \Rightarrow \exists Q'. Q \mapsto (\text{STCal Source Target}) \ast Q' \land Q' \sim_{\cdot \cdot} T < \text{TRel} > R' \)

from this obtain Q' where H1: Q \mapsto (\text{STCal Source Target}) \ast Q' and H2: Q' \sim_{\cdot \cdot} T < \text{TRel} > R'

by blast

assume \( \forall Q'. Q \mapsto (\text{STCal Source Target}) \ast Q' \)

\( \Rightarrow \exists P'. P \mapsto (\text{STCal Source Target}) \ast P' \land P' \sim_{\cdot \cdot} T < \text{TRel} > Q' \)

with H1 obtain P' where H3: P \mapsto (\text{STCal Source Target}) \ast P' and H4: P' \sim_{\cdot \cdot} T < \text{TRel} > Q'

by blast

from H4 H2 have P' \sim_{\cdot \cdot} T < \text{TRel} > R'

by (rule indRelTEQ.trans)

with H3 show \exists P'. P \mapsto (\text{STCal Source Target}) \ast P' \land P' \sim_{\cdot \cdot} T < \text{TRel} > R'

by blast

qed

next

assume \text{bisim: weak-reduction-bisimulation} (\text{indRelTEQ TRel}) (\text{STCal Source Target})

have operational-corresponding \text{TRel}

proof auto

fix S S'

have SourceTerm S \sim_{\cdot \cdot} T < \text{TRel} > TargetTerm ([S])

by (rule indRelTEQ.encR)

moreover assume S \mapsto Source S'

hence SourceTerm S \mapsto (\text{STCal Source Target}) \ast (SourceTerm S')

by (simp add: STCal-steps(1))

ultimately obtain Q' where H1: TargetTerm ([S]) \mapsto (\text{STCal Source Target}) \ast Q'

and H2: SourceTerm S' \sim_{\cdot \cdot} T < \text{TRel} > Q'

using bisim

by blast

from H1 obtain T where H3: [S] \mapsto Target \ast T and H4: T \in T Q'

by (auto simp add: STCal-steps(2))

from eqT have TRel* = TRel

using refl\-trancl\[of TRel\] trancl\-id\[of TRel\]

unfolding equiv\-def refl\-on\-def

by auto
with I2 I4 have \(|[S'], T|) \in TRel
  using \(\text{indRelTEQ-to-TRel}(2)[\text{where } TRel=TRel]\)
  trans-closure-of-TRel-refl-cond[\text{where } TRel=TRel]
  by simp
with I3 show \(\exists T. [S] \mapsto Target\ast T \land ([S'], T) \in TRel\)
  by blast
next
fix \(S \ T\)
have \(\text{SourceTerm } S \sim \{\} T < TRel \ast \text{TargetTerm } ([S])\)
  by (rule \(\text{indRelTEQ.encR}\))
moreover assume \([S] \mapsto Target\ast T\)
hence \(\text{TargetTerm } ([S]) \mapsto (\text{STCal Source Target})\ast (\text{TargetTerm } T)\)
  by (simp add: \(\text{STCal-steps}(2)\))
ultimately obtain \(Q'\) where \(J1: \text{SourceTerm } S \mapsto (\text{STCal Source Target})\ast Q'\)
  and \(J2: Q' \sim \{\} T < TRel \ast \text{TargetTerm } T\)
  using bisim
  by blast
from \(J1\) obtain \(S'\) where \(J3: S \mapsto \text{Source} \ast S'\) and \(J4: S' \in S Q'\)
  by (auto simp add: \(\text{STCal-steps}(1)\))
from eqT have \(TRel^+ = TRel\)
  using reflcl-trancl[of TRel] trancl-id[of TRel]
  unfolding equiv-def refl-on-def
  by auto
with \(J2 J4\) have \([|S'], T|) \in TRel
  using \(\text{indRelTEQ-to-TRel}(2)[\text{where } TRel=TRel]\)
  trans-closure-of-TRel-refl-cond[\text{where } TRel=TRel]
  by blast
with I3 show \(\exists S'. S \mapsto \text{Source} \ast S' \land ([|S'], T|) \in TRel\)
  by blast
qed
moreover have \(\text{weak-reduction-bisimulation } TRel Target\)
proof –
  from eqT have \(TRel^+ = TRel\)
    using reflcl-trancl[of TRel] trancl-id[of TRel]
    unfolding equiv-def refl-on-def
    by auto
  with bisim show \(\text{weak-reduction-bisimulation } TRel Target\)
    using \(\text{indRelTEQ-impl-TRel-is-weak-reduction-bisimulation}[\text{where } TRel=TRel]\)
    by simp
qed
ultimately show \(\text{operational-corresponding } TRel \land \text{weak-reduction-bisimulation } TRel Target\)
  by simp
qed

lemma (in encoding) \(OC\text{-wrt-equivalence-iff-weak-reduction-bisimulation;\)}
fixes \(TRel:: ('\text{procT} \times ' \text{procT}) \text{ set}\)
assumes eqT: \(\text{equivalence } TRel\)
shows \((\text{operational-corresponding } TRel \land \text{weak-reduction-bisimulation } TRel Target) \iff \exists R\text{el.}\)
  \((\forall S. \text{SourceTerm } S, \text{TargetTerm } ([|S|]) \in R \land (\text{TargetTerm } ([|S|]), \text{SourceTerm } S) \in R)\)
  \land \(TRel = \{(T1, T2), \text{TargetTerm } T1, \text{TargetTerm } T2\} \in R\)
  \land \text{trans } R \land \text{weak-reduction-bisimulation } R\text{el} (\text{STCal Source Target})\)
proof (rule iff1, erule conjE)
assume oc: \(\text{operational-corresponding } TRel \text{ and } \text{bisimT: weak-reduction-bisimulation } TRel Target\)
from eqT have \(rt: TRel^+ = TRel\)
  using reflcl-trancl[of TRel] trancl-id[of TRel]
  unfolding equiv-def refl-on-def
  by auto
have \(\forall S. \text{SourceTerm } S \sim \{\} T < TRel \ast \text{TargetTerm } ([|S|]) \land \text{TargetTerm } ([|S|]) \sim \{\} T < TRel \ast \text{SourceTerm } S\)
  by (simp add: \(\text{indRelTEQ.encR \ indRelTEQ.encL}\))
moreover have \(\text{rt have } TRel = \{(T1, T2), \text{TargetTerm } T1, \text{TargetTerm } T2\}\)
  using \(\text{indRelTEQ-to-TRel}(4)[\text{where } TRel=TRel]\)
trans-closure-of-\text{TRel-refl-cond} \text{[where] \text{TRel} = \text{TRel}}
by (auto simp add: indRelTEQ\_target)

moreover have \text{TRel} \text{[indRelTEQ \text{TRel}}
using indRelTEQ\_trans \text{[where] \text{TRel} = \text{TRel}}

unfolding trans-def
by blast

moreover from eqT oc simT
have weak-reduction-bisimulation \text{[indRelTEQ \text{TRel}} (\text{STCal Source Target})
using OC-wrt-equivalence-iff-indRelTEQ\_weak-reduction-bisimulation \text{[where] \text{TRel} = \text{TRel}}
by blast

ultimately
show \( \exists \text{Rel.} (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel} \wedge (\text{TargetTerm} ([S]), \text{SourceTerm} S) \in \text{Rel}) \wedge \text{TRel} = \{(T1, T2), (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel}\} \wedge \text{trans Rel} \wedge \text{weak-reduction-bisimulation \text{Rel} (STCal Source Target)}\)
by blast

next
assume \( \exists \text{Rel.} (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel} \wedge (\text{TargetTerm} ([S]), \text{SourceTerm} S) \in \text{Rel}) \wedge \text{TRel} = \{(T1, T2), (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel}\} \wedge \text{trans Rel} \wedge \text{weak-reduction-bisimulation \text{Rel} (STCal Source Target)}\)
from this obtain Rel \text{where A1:} (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel} \wedge (\text{TargetTerm} ([S]), \text{SourceTerm} S) \in \text{Rel}) \wedge \text{TRel} = \{(T1, T2), (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel}\} \wedge \text{trans \text{Rel}} \wedge \text{weak-reduction-bisimulation \text{Rel} (STCal Source Target)}

and A2: \text{trans Rel} = \{(T1, T2), (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel}\} \text{ and A3: trans Rel}

and A4: \text{weak-reduction-bisimulation \text{Rel} (STCal Source Target)}
by blast

have operational-corresponding \text{TRel}
proof auto
fix S S'
from A1 have (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}
by simp
moreover assume S \rightarrow Source\* S'
hence \text{SourceTerm} S \rightarrow (\text{STCal Source Target})\* (\text{SourceTerm} S')
by (simp add: STCal\_steps(1))

ultimately obtain Q' where B1: \text{TargetTerm} ([S]) \rightarrow (\text{STCal Source Target})\* Q'

and B2: (\text{SourceTerm} S', Q') \in \text{Rel}
using A4
by blast
from B1 obtain T where B3: \text{Target}\* T and B4: T \in T Q'
by (auto simp add: STCal\_steps(2))
from A1 have (\text{TargetTerm} ([S'])), (\text{SourceTerm} S') \in \text{Rel}
by simp
with B2 A3 have (\text{TargetTerm} ([S'])), Q' \in \text{Rel}
unfolding trans-def
by blast
with B4 A2 have ([S'], T) \in \text{TRel}
by simp
with B3 show \( \exists T. ([S] \rightarrow Target\* T \wedge ([S'], T) \in \text{TRel} \)
by blast

next
fix S T
from A1 have (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}
by simp
moreover assume S \rightarrow Target\* T
hence \text{TargetTerm} ([S]) \rightarrow (\text{STCal Source Target})\* (\text{TargetTerm} T)
by (simp add: STCal\_steps(2))
ultimately obtain P' where C1: \text{SourceTerm} S \rightarrow (\text{STCal Source Target})\* P'

and C2: (P', \text{TargetTerm} T) \in \text{Rel}
using A4
by blast
from C1 obtain S' where C3: S \rightarrow Source\* S' and C4: S' \in S P'
by (auto simp add: STCal\_steps(1))
from A1 C4 have \((\text{TargetTerm } ([S'], P')) \in \text{Rel}\) 
by simp
from A3 this C2 have \((\text{TargetTerm } ([S'], \text{TargetTerm } T)) \in \text{Rel}\) 
unfolding trans-def 
by blast
with A2 have \(([S'], T) \in \text{TRel}\) 
by simp
with C3 show \(\exists S'. S \mapsto \text{Source} \ast S' \land ([S'], T) \in \text{TRel}\) 
by blast
qed
moreover have weak-reduction-bisimulation \text{TRel } \text{Target}
proof auto
fix \(TP TQ TP'\)
assume \((TP, TQ) \in \text{TRel}\)
with A2 have \((\text{TargetTerm } TP, \text{TargetTerm } TQ) \in \text{Rel}\) 
by simp
moreover assume \(TP \mapsto \text{Target} \ast TP'\)
hence \(\text{TargetTerm } TP \mapsto (\text{STCal Source Target}) \ast (\text{TargetTerm } TP')\) 
by \((\text{simp add: STCal-steps(2)})\)
ultimately obtain \(Q'\) where \(D1: \text{TargetTerm } TQ \mapsto (\text{STCal Source Target}) \ast Q'\)
and \(D2: ([TP', TQ']) \in \text{TRel}\)
using A4
by blast
from \(D1\) obtain \(TQ'\) where \(D3: TQ \mapsto \text{Target} \ast TQ'\) and \(D4: TQ' \in T Q'\)
by \((\text{auto simp add: STCal-steps(2)})\)
from A2 D2 D4 have \((TP', TQ') \in \text{TRel}\) 
by simp
with \(D3\) show \(\exists TQ'. TQ \mapsto \text{Target} \ast TQ' \land (TP', TQ') \in \text{TRel}\) 
by blast
next
fix \(TP TQ TQ'\)
assume \((TP, TQ) \in \text{TRel}\)
with A2 have \((\text{TargetTerm } TP, \text{TargetTerm } TQ) \in \text{Rel}\) 
by simp
moreover assume \(TQ \mapsto \text{Target} \ast TQ'\)
hence \(\text{TargetTerm } TQ \mapsto (\text{STCal Source Target}) \ast (\text{TargetTerm } TQ')\) 
by \((\text{simp add: STCal-steps(2)})\)
ultimately obtain \(P'\) where \(E1: \text{TargetTerm } TP \mapsto (\text{STCal Source Target}) \ast P'\)
and \(E2: (P', \text{TargetTerm } TQ') \in \text{TRel}\)
using A4
by blast
from \(E1\) obtain \(TP'\) where \(E3: TP \mapsto \text{Target} \ast TP'\) and \(E4: TP' \in T P'\)
by \((\text{auto simp add: STCal-steps(2)})\)
from A2 E2 E4 have \((TP', TQ') \in \text{TRel}\) 
by simp
with \(E3\) show \(\exists TP'. TP \mapsto \text{Target} \ast TP' \land (TP', TQ') \in \text{TRel}\) 
by blast
qed
ultimately show operational-corresponding \text{TRel } \land \text{weak-reduction-bisimulation} \text{TRel Target}
by simp
qed

An encoding is strong operational corresponding w.r.t a strong bisimulation on target terms \text{TRel} iff there exists a relation, like indRelRTPO, that relates at least all source terms and their literal translations, includes \text{TRel}, and is a strong bisimulation. Thus this variant of operational correspondence ensures that source terms and their translations are strong bisimilar.

lemma \((\text{in } \text{encoding})\) SOC-iff-indRelRTPO-is-strong-reduction-bisimulation:
fixes \(\text{TRel} :: ('\text{procT} \times '\text{procT}) \text{ set}\)
shows \((\text{strongly-operational-corresponding} (\text{TRel}^{*}))\)
\land \text{strong-reduction-bisimulation} \((\text{TRel}^{+}) \text{ Target})
\[
\text{proof (rule iffI, erule conjE)}
\]
\begin{itemize}
\item \textbf{assume} \textit{ocorr: strongly-operational-corresponding (TRel*)}
\item \textbf{and bisim: strongly-redution-bisimulation (TRel*) Target}
\item \textbf{hence strongly-redution-simulation (indRelRTPO TRel) (STCal Source Target)}
\item \hspace{1cm} \textbf{using} \textit{SOC-iff-indRelRTPO-is-strong-reduction-simulation[where TRel=TRel]}
\item \hspace{2cm} \textbf{by simp}
\item \textbf{moreover from bisim have strongly-redution-simulation ((TRel*)^{-1}) Target}
\item \hspace{1cm} \textbf{using} \textit{strong-reduction-bisimulations-impl-inverse-is-simulation[where Rel=indRelRTPO TRel]}
\item \hspace{2cm} \textbf{by simp}
\item \textbf{with \textit{ocorr}}
\item \textbf{have strongly-redution-simulation ((indRelRTPO TRel)^{-1}) (STCal Source Target)}
\item \hspace{1cm} \textbf{using} \textit{SOSou-iff-inverse-of-indRelRTPO-is-strong-reduction-simulation[where TRel=TRel]}
\item \hspace{2cm} \textbf{by simp}
\item \textbf{ultimately show strongly-reduction-bisimulation (indRelRTPO TRel) (STCal Source Target)}
\item \hspace{1cm} \textbf{using} \textit{strong-reduction-simulations-impl-bisimulation[where Rel=indRelRTPO TRel]}
\item \hspace{2cm} \textbf{by simp}
\end{itemize}
\textbf{next}
\begin{itemize}
\item \textbf{assume bisim: strongly-redution-bisimulation (indRelRTPO TRel) (STCal Source Target)}
\item \textbf{hence strongly-operational-complete (TRel*)} \land \textit{strongly-redution-simulation (TRel*) Target}
\item \hspace{1cm} \textbf{using} \textit{SOC-iff-indRelRTPO-is-strong-reduction-simulation[where TRel=TRel]}
\item \hspace{2cm} \textbf{by simp}
\item \textbf{moreover from bisim}
\item \textbf{have strongly-redution-simulation ((indRelRTPO TRel)^{-1}) (STCal Source Target)}
\item \hspace{1cm} \textbf{using} \textit{strong-reduction-bisimulations-impl-inverse-is-simulation[where Rel=indRelRTPO TRel]}
\item \hspace{2cm} \textbf{by simp}
\item \textbf{hence strongly-operational-sound (TRel*)} \land \textit{strongly-redution-simulation ((TRel*)^{-1}) Target}
\item \hspace{1cm} \textbf{using} \textit{SOSou-iff-inverse-of-indRelRTPO-is-strong-reduction-simulation[where TRel=TRel]}
\item \hspace{2cm} \textbf{by simp}
\item \textbf{ultimately show strongly-operational-corresponding (TRel*)}
\item \hspace{1cm} \land \textit{strongly-redution-bisimulation (TRel*) Target}
\item \hspace{2cm} \textbf{using} \textit{strong-reduction-simulations-impl-bisimulation[where Rel=TRel]}
\item \hspace{2cm} \textbf{by simp}
\end{itemize}
\textbf{qed}

\textbf{lemma (in encoding)} \textit{SOC-iff-strong-reduction-bisimulation:}
\begin{itemize}
\item \textbf{fixes TRel :: ('procT × 'procT) set}
\item \textbf{shows (strongly-operational-corresponding (TRel*)}
\item \hspace{1cm} \land \textit{strong-reduction-bisimulation (TRel*) Target)}
\item \hspace{1cm} \begin{align*}
&= (\exists \textit{Rel}. (\forall S. (\textit{SourceTerm S, TargetTerm ([S]])) \in TRel)
\land (\forall T1 T2. (T1, T2) \in TRel \rightarrow (\textit{TargetTerm T1, TargetTerm T2}) \in TRel)
\land (\forall S. (S, (\textit{SourceTerm S, TargetTerm T}) \in TRel \rightarrow ([S], T) \in TRel^*))
\land \textit{strong-reduction-bisimulation Rel (STCal Source Target)})
\end{align*}
\end{itemize}
\textbf{proof (rule iffI, erule conjE)}
\begin{itemize}
\item \textbf{have} \begin{align*}
&\forall S. (\textit{SourceTerm S, TargetTerm ([S]])) \in \textit{indRelRTPO TRel}
\end{align*}
\item \hspace{1cm} \textbf{by (simp add: indRelRTPO.encoding)}
\item \textbf{moreover have} \begin{align*}
&\forall T1 T2. (T1, T2) \in TRel \rightarrow \textit{TargetTerm T1} \lesssim \textit{RT} <\textit{TRel}> \textit{TargetTerm T2}
\end{align*}
\item \hspace{2cm} \textbf{by (simp add: indRelRTPO.target)}
\item \textbf{moreover have} \begin{align*}
&\forall T1 T2. \textit{TargetTerm T1} \lesssim \textit{RT} <\textit{TRel}> \textit{TargetTerm T2} \rightarrow (T1, T2) \in TRel^+
\end{align*}
\item \hspace{2cm} \textbf{using} \begin{align*}
&\textit{indRelRTPO-to-TRel(4)[where TRel=TRel]}
\end{align*}
\item \hspace{2cm} \textbf{by simp}
\item \textbf{moreover have} \begin{align*}
&\forall S T. \textit{SourceTerm S} \lesssim \textit{RT} <\textit{TRel}> \textit{TargetTerm T} \rightarrow ([S], T) \in TRel^*
\end{align*}
\item \hspace{2cm} \textbf{using} \begin{align*}
&\textit{indRelRTPO-to-TRel(2)[where TRel=TRel]} \textit{trans-closure-of-TRel-refl-cond}
\end{align*}
\item \hspace{2cm} \textbf{by simp}
\item \textbf{moreover assume} \textit{strongly-operational-corresponding (TRel*)}
\item \hspace{1cm} \textbf{and strongly-redution-bisimulation (TRel*) Target}
\item \hspace{2cm} \textbf{using} \begin{align*}
&\textit{SOC-iff-indRelRTPO-is-strong-reduction-bisimulation[where TRel=TRel]}
\end{align*}
\item \hspace{2cm} \textbf{by simp}
\item \textbf{ultimately show} \begin{align*}
&\exists \textit{Rel}. (\forall S. (\textit{SourceTerm S, TargetTerm ([S]])) \in TRel)
\end{align*}
\( \forall T_1 T_2. (T_1, T_2) \in TRel \rightarrow (\text{TargetTerm} T_1, \text{TargetTerm} T_2) \in \text{Rel} \)
\( \forall T_1 T_2. (\text{TargetTerm} T_1, \text{TargetTerm} T_2) \in \text{Rel} \rightarrow (T_1, T_2) \in TRel^* \)
\( \forall S T. (\text{SourceTerm} S, \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in TRel^* \)
\( \text{strong-reduction-bisimulation} \ \text{Rel} \ (\text{STCal Source Target}) \)

by blast

next
assume \( \exists \text{Rel}. (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}) \)
\( \land (\forall T_1 T_2. (T_1, T_2) \in TRel \rightarrow (\text{TargetTerm} T_1, \text{TargetTerm} T_2) \in \text{Rel}) \)
\( \land (\forall T_1 T_2. (\text{TargetTerm} T_1, \text{TargetTerm} T_2) \in \text{Rel} \rightarrow (T_1, T_2) \in TRel^*) \)
\( \land (\forall S T. (\text{SourceTerm} S, \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in TRel^*) \)
\( \text{strong-reduction-bisimulation} \ \text{Rel} \ (\text{STCal Source Target}) \)

from this obtain \( \text{Rel} \) where \( A_1: \forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel} \)
and \( A_2: \forall T_1 T_2. (T_1, T_2) \in TRel \rightarrow (\text{TargetTerm} T_1, \text{TargetTerm} T_2) \in \text{Rel} \)
and \( A_3: \forall T_1 T_2. (\text{TargetTerm} T_1, \text{TargetTerm} T_2) \in \text{Rel} \rightarrow (T_1, T_2) \in TRel^* \)
and \( A_4: \forall S T. (\text{SourceTerm} S, \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in TRel^* \)
and \( A_5: \text{strong-reduction-bisimulation} \ \text{Rel} \ (\text{STCal Source Target}) \)

by blast

hence \( \text{strongly-operational-complete} \ (TRel^*) \)
\( \land \text{strong-reduction-simulation} \ (TRel^*) \ \text{Target} \)

using \( \text{SOCom-iff-strong-reduction-simulation\ where TRel=TRel} \)

by blast

moreover from \( A_5 \) have \( \text{strongly-operational-sound} \ (TRel^*) \)
\( \land \text{strong-reduction-simulation} \ ((TRel^*)^{-1}) \ \text{Target} \)

using \( \text{SOSou-iff-strong-reduction-simulation\ where TRel=TRel} \)

by simp

with \( A_1 A_2 A_3 A_4 \) have \( \text{strongly-operational-sound} \ (TRel^*) \)
\( \land \text{strong-reduction-simulation} \ ((TRel^*)^{-1}) \ \text{Target} \)

using \( \text{SOSou-iff-strong-reduction-simulation\ where TRel=TRel} \)

by blast

ultimately show \( \text{strongly-operational-corresponding} \ (TRel^*) \)
\( \land \text{strong-reduction-bisimulation} \ (TRel^*) \ \text{Target} \)

using \( \text{strong-reduction-simulations-impl-bisimulation\ where TRel=TRel^*} \)

by simp

qed

Lemma (in encoding) \( \text{SOC-wrt-preorder-iff-strong-reduction-bisimulation:} \)

\( \text{Proof (rule iffI,erule conjE,erule conjE,erule conjE)} \)

assume \( A_1: \text{strongly-operational-complete} \ TRel \ \text{and} \ A_2: \text{strongly-operational-sound} \ TRel \)
and \( A_3: \text{preorder} \ TRel \ \text{and} \ A_4: \text{strong-reduction-bisimulation} \ TRel \ \text{Target} \)

from \( A_3 \) have \( A_5: TRel^* = TRel \)

using \( \text{transcl-id[of TRel]} \)

unfolding \( \text{preorder-on-def} \)

by blast

with \( A_3 \) have \( TRel^* = TRel \)

using \( \text{reflcl-trancl[of TRel]} \)

unfolding \( \text{preorder-on-def refl-on-def} \)

by blast

with \( A_1 A_2 \) have \( \text{strongly-operational-corresponding} \ (TRel^*) \)

by simp

moreover from \( A_4 A_5 \) have \( \text{strong-reduction-bisimulation} \ (TRel^*) \ \text{Target} \)

by simp

ultimately

have \( \text{strong-reduction-bisimulation} \ (\text{indRelRTPO \ TRel}) \ (\text{STCal Source Target}) \)

using \( \text{SOC-iff-indRelRTPO-is-strong-reduction-bisimulation\ where TRel=TRel} \)
by blast

moreover have $\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{indRelRTPO } \mathit{TRel}$
by (simp add: indRelRTPO_encR)

moreover
have $\mathit{TRel} = \{(T1, T2), (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{indRelRTPO } \mathit{TRel}\}$
proof auto
  fix $\mathit{TP} \mathit{TQ}$
  assume $(\mathit{TP}, \mathit{TQ}) \in \mathit{TRel}$
  thus $\text{TargetTerm } \mathit{TP} \preceq_\mathit{RT} \mathit{TQ}$
  by (rule indRelRTPO_target)
next
  fix $\mathit{TP} \mathit{TQ}$
  assume $\text{TargetTerm } \mathit{TP} \preceq_\mathit{RT} \mathit{TQ}$
  with $A3$ show $(\mathit{TP}, \mathit{TQ}) \in \mathit{TRel}$
    using indRelRTPO-to-TRel(4)[where $\mathit{TRel} = \mathit{TRel}$]
    unfolding preorder-on-def refl-on-def
    by blast
qed

moreover from $A3$
have $\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \text{indRelRTPO } \mathit{TRel} \longrightarrow ([S], T) \in \mathit{TRel}^+$
  using indRelRTPO-to-TRel(2)[where $\mathit{TRel} = \mathit{TRel}$]
  unfolding preorder-on-def refl-on-def
  by blast

with $A3$ have $\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \text{indRelRTPO } \mathit{TRel} \longrightarrow ([S], T) \in \mathit{TRel}$
  using trancl-id[of $\mathit{TRel}$]
  unfolding preorder-on-def
  by blast

moreover have $\text{refl } (\text{indRelRTPO } \mathit{TRel})$
  unfolding preorder-on-def
  by (simp add: indRelRTPO-refl)

moreover have $\text{trans } (\text{indRelRTPO } \mathit{TRel})$
  unfolding preorder-on-def
  trans-def
  unfolding preorder-on-def
  by blast

ultimately show $\exists \mathit{Rel}. (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S]))) \in \mathit{Rel}$
  \& $\mathit{TRel} = \{(T1, T2), (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \mathit{Rel}\}$
  \& $\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \mathit{Rel} \longrightarrow ([S], T) \in \mathit{TRel}$
  \& preorder $\mathit{Rel}$
  \& strong-reduction-bisimulation $\mathit{Rel} (\text{STCal Source Target})$
  unfolding preorder-on-def
  by blast
next
assume $\exists \mathit{Rel}. (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S]))) \in \mathit{Rel}$
  \& $\mathit{TRel} = \{(T1, T2), (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \mathit{Rel}\}$
  \& $\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \mathit{Rel} \longrightarrow ([S], T) \in \mathit{TRel}$
  \& preorder $\mathit{Rel}$
  \& strong-reduction-bisimulation $\mathit{Rel} (\text{STCal Source Target})$
from this obtain $\mathit{Rel}$ where $B1$: $\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \mathit{Rel}$
and $B2$: $\mathit{TRel} = \{(T1, T2), (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \mathit{Rel}\}$
and $B3$: $\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \mathit{Rel} \longrightarrow ([S], T) \in \mathit{TRel}$ and $B4$: preorder $\mathit{Rel}$
and $B5$: strong-reduction-bisimulation $\mathit{Rel} (\text{STCal Source Target})$
by blast
from $B2$ $B4$ have $B6$: $\text{refl } \mathit{TRel}$
  unfolding preorder-on-def refl-on-def
  by blast
from $B2$ $B4$ have $B7$: $\text{trans } \mathit{TRel}$
  unfolding trans-def preorder-on-def
  by blast
hence $B8$: $\mathit{TRel}^+ = \mathit{TRel}$
by (rule trancl-id)
with \( B6 \) have \( B9: \text{TRel}^* = \text{TRel} \)
  using refl-trancl[of \text{TRel}]
  unfolding refl-on-def
by blast

with \( B3 \) have \( \forall S.T. (\text{SourceTerm} S, \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}^* \)
by simp

moreover from \( B2 B8 \) have \( \forall T1 T2. (T1, T2) \in \text{TRel} \rightarrow (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel} \)
and \( \forall T1 T2. (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel} \rightarrow (T1, T2) \in \text{TRel}^* \)
by auto

ultimately have \( \exists \text{Rel}. (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}) \)
\( \land (\forall T1 T2. (T1, T2) \in \text{TRel} \rightarrow (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel}) \)
\( \land (\forall T1 T2. (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel} \rightarrow (T1, T2) \in \text{TRel}^*) \)
\( \land \text{strong-reduction-bisimulation} \text{Rel} (\text{STCal} \text{Source} \text{Target}) \)
using \( B1 B5 \)
by blast

hence \( \text{strongly-operational-corresponding} (\text{TRel}^*) \land \text{strong-reduction-bisimulation} (\text{TRel}^+) \) \( \text{Target} \)
using \( \text{SOC-if strongly-reduction-bisimulation}[\text{where} \text{TRel} = \text{TRel}] \)
by simp

with \( B8 B9 \) have \( \text{strongly-operational-corresponding} \text{TRel} \land \text{strong-reduction-bisimulation} \text{TRel} \text{Target} \)
by simp

moreover from \( B6 B7 \) have \( \text{preorder} \text{TRel} \)

unfolding \( \text{preorder-on-def} \)
by blast

ultimately show \( \text{strongly-operational-corresponding} \text{TRel} \land \text{preorder} \text{TRel} \)
\( \land \text{strong-reduction-bisimulation} \text{TRel} \text{Target} \)
by blast

qed

lemma (in encoding) \( \text{SOC-wrt-TRel-iff-strong-reduction-bisimulation:} \)
shows \( (\exists \text{TRel. strongly-operational-corresponding} (\text{TRel}^*) \land \text{strong-reduction-bisimulation} (\text{TRel}^+) \) \( \) \( \text{Target} \)
\( = (\exists \text{Rel}. (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}) \land (\forall T1 T2. (T1, T2) \in \text{TRel} \rightarrow (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel}) \rightarrow (T1, T2) \in \text{TRel}^*) \land \text{strong-reduction-bisimulation} \text{Rel} (\text{STCal} \text{Source} \text{Target}) \)
proof (rule iffI)
assume \( (\exists \text{TRel. strongly-operational-corresponding} (\text{TRel}^*) \land \text{strong-reduction-bisimulation} (\text{TRel}^+) \) \( \text{Target} \)
from this obtain \( \text{TRel where} \text{strongly-operational-corresponding} (\text{TRel}^*) \land \text{strong-reduction-bisimulation} (\text{TRel}^+) \) \( \text{Target} \)
by blast
hence \( \text{strong-reduction-bisimulation} (\text{indRelRTPO} \text{TRel}) (\text{STCal} \text{Source} \text{Target}) \)
using \( \text{SOC-if-indRelRTPO-is-strong-reduction-bisimulation}[\text{where} \text{TRel} = \text{TRel}] \)
by simp

moreover have \( (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{indRelRTPO} \text{TRel} \)
by (simp add: \text{indRelRTPO_EncR})

moreover have \( (\forall T1 T2. (T1, T2) \in \text{indRelRTPO} \text{TRel} \rightarrow (\text{TargetTerm} ([S]), \text{TargetTerm} T) \in \text{indRelRTPO} \text{TRel}^+) \)
using \( \text{indRelRTPO-relates-source-target}[\text{where} \text{TRel} = \text{TRel}] \)
by simp

ultimately show \( (\exists \text{Rel}. (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}) \land (\forall T1 T2. (T1, T2) \in \text{Rel} \rightarrow (\text{TargetTerm} ([S]), \text{TargetTerm} T) \in \text{Rel}^*) \land \text{strong-reduction-bisimulation} \text{Rel} (\text{STCal} \text{Source} \text{Target}) \)
by blast

next
assume \( (\exists \text{Rel}. (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}) \land (\forall T1 T2. (T1, T2) \in \text{Rel} \rightarrow (\text{TargetTerm} ([S]), \text{TargetTerm} T) \in \text{Rel}^*) \land \text{strong-reduction-bisimulation} \text{Rel} (\text{STCal} \text{Source} \text{Target}) \)
from this obtain \( \text{Rel where} \text{A1:} \forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel} \)
and $A2: \forall S T. (SourceTerm S, TargetTerm T) \in Rel$

$\rightarrow (TargetTerm ([S]), TargetTerm T) \in Rel$.

and $A3: \text{strong-reduction-bisimulation } Rel \ (STCal \ Source \ Target)$

by blast

from $A2$ obtain $TRel$ where $\forall T1 T2. (T1, T2) \in TRel \rightarrow (TargetTerm T1, TargetTerm T2) \in Rel$

and $\forall T1 T2. (TargetTerm T1, TargetTerm T2) \in Rel \rightarrow (T1, T2) \in TRel^+$

and $\forall S T. (SourceTerm S, TargetTerm T) \in Rel \rightarrow ([S], T) \in TRel^+$

using target-relation-from-source-target-relation[where $Rel=\text{Rel}$]

by blast

with $A1 \ A3$ have $\exists \text{Rel}. (\forall S. (SourceTerm S, TargetTerm ([S])) \in Rel)$

$\land (\forall T1 T2. (T1, T2) \in TRel \rightarrow (TargetTerm T1, TargetTerm T2) \in Rel)$

$\land (\forall T1 T2. (TargetTerm T1, TargetTerm T2) \in Rel \rightarrow (T1, T2) \in TRel^+)$

$\land (\forall S T. (SourceTerm S, TargetTerm T) \in Rel \rightarrow ([S], T) \in TRel^+)$

$\land \text{strong-reduction-bisimulation } Rel \ (STCal \ Source \ Target)$

by blast

hence strongly-operational-corresponding (TRel$^+$)

$\land \text{strong-reduction-bisimulation } (TRel^+)$ Target

using SOC-iff-strong-reduction-bisimulation[where $TRel=\text{TRel}$]

by simp

thus $\exists TRel. \text{strongly-operational-corresponding } (TRel^+)$

$\land \text{strong-reduction-bisimulation } (TRel^+)$ Target

by blast

qed

lemma (in encoding) SOC-wrt-equivalence-iff-indRelTEQ-strong-reduction-bisimulation:

fixes $TRel :: ('procT \times 'procT) \text{ set}$

assumes $eqT: \text{equivalence } TRel$

shows (strong-operational-corresponding $TRel \land \text{strong-reduction-bisimulation } (\text{TRel Target})$

$\iff \text{strong-reduction-bisimulation } (\text{indRelTEQ Target})$ $\ (STCal \ Source \ Target)$

proof (rule ifI, erule conjE)

assume $oc: \text{strongly-operational-corresponding } TRel$

and $bismT: \text{strong-reduction-bisimulation } TRel \ Source \ Target$

show $\text{strong-reduction-bisimulation } (\text{indRelTEQ TRel})$ $\ (STCal \ Source \ Target)$

proof auto

fix $P \ Q \ P'$

assume $P \sim [\cdot] T < TRel > Q \land \ P \iff (STCal \ Source \ Target) \ P'$

thus $\exists Q'. Q \iff (STCal \ Source \ Target) \ Q' \land \ P' \sim [\cdot] T < TRel > Q'$

proof (induct arbitrary: $P'$)

case $(\text{encR } S)$

assume $\text{SourceTerm } S \rightarrow (STCal \ Source \ Target) \ P'$

from this obtain $S'$ where $A1: S \\rightarrow \text{Source } S'$ and $A2: S' \in S \ P'$

by (auto simp add: STCal-step(1))

from $A1 \ oc$ obtain $T$ where $A3: [S] \rightarrow Target \ T$ and $A4: ([S'], T) \in TRel$

by blast

from $A3$ have $\text{TargetTerm } ([S]) \rightarrow (STCal \ Source \ Target) \ (TargetTerm T)$

by (simp add: STCal-step(2))

moreover have $P' \sim [\cdot] T < TRel > TargetTerm T$

proof

from $A2$ have $P' \sim [\cdot] T < TRel > TargetTerm ([S'])$

by (simp add: indRelTEQ.encR)

moreover from $A4$ have $\text{TargetTerm } ([S']) \sim [\cdot] T < TRel > TargetTerm T$

by (rule indRelTEQ.target)

ultimately show $P' \sim [\cdot] T < TRel > TargetTerm T$

by (rule indRelTEQ.trans)

qed

ultimately show $\exists Q'. \text{TargetTerm } ([S]) \rightarrow (STCal \ Source \ Target) \ Q' \land \ P' \sim [\cdot] T < TRel > Q'$

by blast

next

case $(\text{encL } S)$

assume $\text{TargetTerm } ([S]) \rightarrow (STCal \ Source \ Target) \ P'$

from this obtain $T$ where $B1: [S] \rightarrow Target \ T$ and $B2: T \in T \ P'$
by (auto simp add: STCal-step(2))
from B1 oc obtain S' where B3: S' ⟷ Source S' and B4: (\([S'], T\) ∈ TRel
by blast
from B3 have SourceTerm S ⟷ (STCal Source Target) (SourceTerm S')
by (simp add: STCal-step(I))
moreover have P' ~[\([\cdot]\)]T<Rel> SourceTerm S'
proof
  from B4 eqT have (T, \([S']\) ∈ TRel
    unfolding equiv-def sym-def
    by blast
  with B2 have P' ~[\([\cdot]\)]T<Rel> TargetTerm \([\([S']\)\]
    by (simp add: indRelTEQ.target)
  moreover have TargetTerm \([\([S']\)\] ~[\([\cdot]\)]T<Rel> SourceTerm S'
    by (rule indRelTEQ.encl)
  ultimately show P' ~[\([\cdot]\)]T<Rel> SourceTerm S'
    by (rule indRelTEQ.trans)
qed
ultimately show ∃ Q'. SourceTerm S ⟷ (STCal Source Target) Q' ∧ P' ~[\([\cdot]\)]T<Rel> Q'
by blast
next
case (target T1 T2)
assume TargetTerm T1 ⟷ (STCal Source Target) P'
from this obtain T1' where C1: T1' ⟷ Target T1' and C2: T1' ∈ T P'
  by (auto simp add: STCal-step(2))
assume (T1, T2) ∈ TRel
with C1 bisimT obtain T2' where C3: T2' ⟷ Target T2' and C4: (T1', T2') ∈ TRel
  by blast
from C3 have TargetTerm T2 ⟷ (STCal Source Target) (TargetTerm T2')
  by (simp add: STCal-step(I))
moreover from C2 C4 have P' ~[\([\cdot]\)]T<Rel> TargetTerm T2'
  by (simp add: indRelTEQ.target)
ultimately show ∃ Q'. TargetTerm T2 ⟷ (STCal Source Target) Q' ∧ P' ~[\([\cdot]\)]T<Rel> Q'
  by blast
next
case (trans P Q R)
assume P ⟷ (STCal Source Target) P'
  and \(\exists Q'. Q ⟷ (STCal Source Target) Q' ∧ P' ~[\([\cdot]\)]T<Rel> Q'\)
from this obtain Q' where D1: Q ⟷ (STCal Source Target) Q' and D2: P' ~[\([\cdot]\)]T<Rel> Q'
  by blast
assume \(\exists Q'. Q ⟷ (STCal Source Target) Q' \)
  \(\exists R'. R ⟷ (STCal Source Target) R' ∧ Q' ~[\([\cdot]\)]T<Rel> R'\)
with D1 obtain R' where D3: R ⟷ (STCal Source Target) R' and D4: Q' ~[\([\cdot]\)]T<Rel> R'
  by blast
from D2 D4 have P' ~[\([\cdot]\)]T<Rel> R'
  by (rule indRelTEQ.trans)
with D3 show \(\exists R'. R ⟷ (STCal Source Target) R' ∧ P' ~[\([\cdot]\)]T<Rel> R'\)
  by blast
qed
next
fix P Q Q'
assume P ~[\([\cdot]\)]T<Rel> Q and Q ⟷ (STCal Source Target) Q'
thus \(\exists P', P ⟷ (STCal Source Target) P' ∧ P' ~[\([\cdot]\)]T<Rel> Q'\)
proof (induct arbitrary: Q')
case (encR S)
assume TargetTerm \([\([S]\)\] ⟷ (STCal Source Target) Q'
from this obtain T where E1: \([\([S]\)\] ⟷ Target T and E2: T ∈ T Q'
  by (auto simp add: STCal-step(2))
from E1 oc obtain S' where E3: S ⟷ Source S' and E4: \([\([S]\), T\) ∈ TRel
  by blast
from E3 have SourceTerm S ⟷ (STCal Source Target) (SourceTerm S')
by (simp add: STCal-step(1))
moreover have SourceTerm S' ~[\cdot] T \,<\, TRel \, Q'
proof -
  have SourceTerm S' ~[\cdot] T \,<\, TRel \, TargetTerm ([S'])
    by (rule indRelTEQ encR)
  moreover from E2 E4 have TargetTerm ([S']) ~[\cdot] T \,<\, TRel \, Q'
    by (simp add: indRelTEQ.target)
  ultimately show SourceTerm S' ~[\cdot] T \,<\, TRel \, Q'
    by (rule indRelTEQ.trans)
qed
ultimately show \exists \, P'. SourceTerm S \,\longmapsto\,(STCal \, Source \, Target) \, P' \,\land\, P' \sim[\cdot] T \,<\, TRel \, Q'
  by blast
next
case (encL S)
  assume SourceTerm S \,\longmapsto\,(STCal \, Source \, Target) \, Q'
  from this obtain S' where F1: S \,\longmapsto\, Source \, S' \,\land\, F2: S' \,\in\, S \, Q'
    by (auto simp add: STCal-step(1))
  from F1 oc obtain T where F3: [S] \,\longmapsto\, Target \, T \,\land\, F4: ([S'], T) \,\in\, TRel
    by blast
  from F3 have TargetTerm ([S]) \,\longmapsto\,(STCal \, Source \, Target) \,(TargetTerm \, T)
    by (simp add: STCal-step(2))
  moreover have TargetTerm T \sim[\cdot] T \,<\, TRel \, Q'
    proof -
      from F4 eqT have (T, [S']) \,\in\, TRel
        unfolding equiv-def sym-def
      by blast
      hence TargetTerm T \sim[\cdot] T \,<\, TRel \, TargetTerm ([S'])
        by (rule indRelTEQ.target)
      moreover from F2 have TargetTerm ([S']) \sim[\cdot] T \,<\, TRel \, Q'
        by (simp add: indRelTEQ.encL)
      ultimately show TargetTerm T \sim[\cdot] T \,<\, TRel \, Q'
        by (rule indRelTEQ.trans)
    qed
    ultimately show \exists \, P'. TargetTerm ([S]) \,\longmapsto\,(STCal \, Source \, Target) \, P' \,\land\, P' \sim[\cdot] T \,<\, TRel \, Q'
      by blast
next
case (target T1 T2)
  assume TargetTerm T2 \,\longmapsto\,(STCal \, Source \, Target) \, Q'
  from this obtain T2' where G1: T2 \,\longmapsto\, Target \, T2' \,\land\, G2: T2' \,\in\, T \, Q'
    by (auto simp add: STCal-step(2))
  assume (T1, T2') \,\in\, TRel
  with G1 bisimT obtain T1' where G3: T1 \,\longmapsto\, Target \, T1' \,\land\, G4: (T1', T2') \,\in\, TRel
    by blast
  from G3 have TargetTerm T1 \,\longmapsto\,(STCal \, Source \, Target) \,(TargetTerm \, T1')
    by (simp add: STCal-step(2))
  moreover from G2 G4 have TargetTerm T1' \sim[\cdot] T \,<\, TRel \, Q'
    by (simp add: indRelTEQ.target)
  ultimately show \exists \, P'. TargetTerm T1 \,\longmapsto\,(STCal \, Source \, Target) \, P' \,\land\, P' \sim[\cdot] T \,<\, TRel \, Q'
    by blast
next
case (trans P Q R R')
  assume R \,\longmapsto\,(STCal \, Source \, Target) \, R'
  and \^[R']. R \,\longmapsto\,(STCal \, Source \, Target) \, R'
    \implies \exists \, Q'. Q \,\longmapsto\,(STCal \, Source \, Target) \, Q' \,\land\, Q' \sim[\cdot] T \,<\, TRel \, R'
  from this obtain Q' where H1: Q \,\longmapsto\,(STCal \, Source \, Target) \, Q' \,\land\, H2: Q' \sim[\cdot] T \,<\, TRel \, R'
    by blast
  assume \^[Q']. Q \,\longmapsto\,(STCal \, Source \, Target) \, Q'
    \implies \exists \, P'. P \,\longmapsto\,(STCal \, Source \, Target) \, P' \,\land\, P' \sim[\cdot] T \,<\, TRel \, Q'
  with H1 obtain P' where H3: P \,\longmapsto\,(STCal \, Source \, Target) \, P' \,\land\, H4: P' \sim[\cdot] T \,<\, TRel \, Q'
    by blast
  from H4 H2 have P' \sim[\cdot] T \,<\, TRel \, R'

proof

with H3 show \( \exists P'. P \mapsto (S \text{Cal Source Target}) \) \( P' \land P' \sim^{\Sigma} T \mapsto \text{TRel} > R' \)

by (rule indRelTEQ.trans)

by blast

qed

qed

next

assume bisim: strong-reduction-bisimulation (indRelTEQ TRel) (S\text{Cal Source Target})

have strongly-operational-corresponding TRel

proof auto

fix \( S S' \)

have SourceTerm \( S \sim^{\Sigma} T \mapsto \text{TRel} > \text{TargetTerm} ([S]) \)

by (rule indRelTEQ.encR)

moreover assume \( S \mapsto \text{Source S} ' \)

hence SourceTerm \( S \mapsto (S \text{Cal Source Target}) (\text{SourceTerm S}') \)

by (simp add: S\text{Cal-step(1)})

ultimately obtain \( Q' \) where \( I1: \text{TargetTerm} ([S]) \mapsto (S \text{Cal Source Target}) Q' \)

and \( I2: \text{SourceTerm S}' \sim^{\Sigma} T \mapsto \text{TRel} > Q' \)

using bisim

by blast

from \( I1 \) obtain \( T \) where \( I3: [S] \mapsto \text{TargetTerm T and I4: T} \in T \equiv Q' \)

by (auto simp add: STCal-step(2))

from eqT have \( \text{TRel}^* = \text{TRel} \)

using reflcl-trancl[of \text{TRel}] trancl-id[of \text{TRel}]

unfolding equiv-def refl-on-def

by auto

with \( I2 \) \( I4 \) have \( ([S]', T) \in \text{TRel} \)

using indRelTEQ-to-TRel(2)[where \text{TRel}=\text{TRel}]

trans-closure-of-TRel-refl-cond[where \text{TRel}=\text{TRel}]

by simp

with \( I3 \) show \( \exists T. [S] \mapsto \text{TargetTerm T and (}[S]', T) \in \text{TRel} \)

by blast

next

fix \( S T \)

have SourceTerm \( S \sim^{\Sigma} T \mapsto \text{TRel} > \text{TargetTerm} ([S]) \)

by (rule indRelTEQ.encR)

moreover assume \( [S] \mapsto \text{TargetTerm T} \)

hence \( \text{TargetTerm} ([S]) \mapsto (S \text{Cal Source Target}) (\text{TargetTerm T}) \)

by (simp add: S\text{Cal-step(2)})

ultimately obtain \( Q' \) where \( J1: \text{SourceTerm S} \mapsto (S \text{Cal Source Target}) Q' \)

and \( J2: Q' \sim^{\Sigma} T \mapsto \text{TRel} > \text{TargetTerm T} \)

using bisim

by blast

from \( J1 \) obtain \( S' \) where \( J3: S \mapsto \text{Source S'} and J4: S' \in S \equiv Q' \)

by (auto simp add: STCal-step(1))

from eqT have \( \text{TRel}^* = \text{TRel} \)

using reflcl-trancl[of \text{TRel}] trancl-id[of \text{TRel}]

unfolding equiv-def refl-on-def

by auto

with \( J2 \) \( J4 \) have \( ([S]', T) \in \text{TRel} \)

using indRelTEQ-to-TRel(2)[where \text{TRel}=\text{TRel}]

trans-closure-of-TRel-refl-cond[where \text{TRel}=\text{TRel}]

by blast

with \( J3 \) show \( \exists S'. S \mapsto \text{Source S'} \and ([S]', T) \in \text{TRel} \)

by blast

qed

moreover have strong-reduction-bisimulation \( \text{TRel Target} \)

proof

from eqT have \( \text{TRel}^* = \text{TRel} \)

using reflcl-trancl[of \text{TRel}] trancl-id[of \text{TRel}]

unfolding equiv-def refl-on-def

by auto
With bisim show strong-reduction-bisimulation TRel Target 
  using indRelTEQ-impl-TRel-is-strong-reduction-bisimulation[where TRel = TRel]
  by simp
qed
ultimately
show strongly-operational-corrresponding TRel \land strong-reduction-bisimulation TRel Target 
  by simp
qed

Lemma (in encoding) SOC-wrt-equivalence-iff-strong-reduction-bisimulation:
  fixes TRel :: ('procT \times 'procT) set 
  assumes eqT: equivalence TRel 
  shows (strongly-operational-corrresponding TRel \land strong-reduction-bisimulation TRel Target)
    \leftrightarrow (\exists Rel.
      (\forall S. (SourceTerm S, TargetTerm (\[ S \])) \in Rel \land (TargetTerm (\[ S \]), SourceTerm S) \in Rel)
      \land TRel = \{(T1, T2), (TargetTerm T1, TargetTerm T2) \in Rel\} 
      \land trans Rel \land strong-reduction-bisimulation Rel (STCal Source Target)
    )
  proof (rule ifI,erule conjE)
    assume oc: strongly-operational-corrresponding TRel 
    and bisimT: strong-reduction-bisimulation TRel Target 
    from eqT have rt: TRel' = TRel 
      using refl-trancl[of TRel], trancl-id[of TRel]
      unfolding equiv-def refl-on-def 
      by auto
    have \forall S. SourceTerm S \sim[\[ T< TRel> TargetTerm (\[ S \]) \land TargetTerm (\[ S \]) \sim[\[ T< TRel> SourceTerm S 
      by (simp add: indRelTEQ.encR indRelTEQ.encL)
    moreover from rt have TRel = \{(T1, T2), (TargetTerm T1, TargetTerm T2) \sim[\[ T< TRel> TargetTerm T2\}
      using indRelTEQ-to-TRel(4)[where TRel = TRel]
      trans-closure-of-TRel-refl-cond[where TRel = TRel]
      by (auto simp add: indRelTEQ.target)
    moreover have trans (indRelTEQ TRel)
      using indRelTEQ,trans[where TRel = TRel]
      unfolding trans-def 
      by blast
    moreover from eqT oc bisimT
    have strong-reduction-bisimulation (indRelTEQ TRel) (STCal Source Target)
      using SOC-wrt-equivalence-iff-indRelTEQ-strong-reduction-bisimulation[where TRel = TRel]
      by blast
    ultimately
    show \exists Rel. (\forall S. (SourceTerm S, TargetTerm (\[ S \])) \in Rel \land (TargetTerm (\[ S \]), SourceTerm S) \in Rel)
      \land TRel = \{(T1, T2), (TargetTerm T1, TargetTerm T2) \in Rel\} \land trans Rel 
      \land strong-reduction-bisimulation Rel (STCal Source Target)
      by blast
  next
  assume \exists Rel. (\forall S. (SourceTerm S, TargetTerm (\[ S \])) \in Rel 
    \land (TargetTerm (\[ S \]), SourceTerm S) \in Rel)
    \land TRel = \{(T1, T2), (TargetTerm T1, TargetTerm T2) \in Rel\} \land trans Rel 
    \land strong-reduction-bisimulation Rel (STCal Source Target)
  from this obtain Rel where A1: \forall S. (SourceTerm S, TargetTerm (\[ S \])) \in Rel 
    \land (TargetTerm (\[ S \]), SourceTerm S) \in Rel
    and A2: TRel = \{(T1, T2), (TargetTerm T1, TargetTerm T2) \in Rel\} and A3: trans Rel 
    and A4: strong-reduction-bisimulation Rel (STCal Source Target)
    by blast
  have strongly-operational-corrresponding TRel 
    proof auto
    fix S S'
    from A1 have (SourceTerm S, TargetTerm (\[ S \])) \in Rel 
      by simp
    moreover assume S \mapsto Source S'
    hence SourceTerm S \mapsto (STCal Source Target) (SourceTerm S') 
      by (simp add: STCal-step(1))
ultimately obtain \( Q' \) where \( B1: \TargetTerm ([S]) \land\rightarrow\STCal\ Source\ Target \) \( Q' \)
and \( B2: (\SourceTerm S', Q') \in \Rel \)

using \( A4 \)
by blast
from \( B1 \) obtain \( T \) where \( B3: [S] \land\rightarrow\Target T \) and \( B4: T \in T Q' \)
by (auto simp add: \STCal-step(2))
from \( A1 \) have \( (\TargetTerm ([S']), \SourceTerm S') \in \Rel \)
by simp
with \( B2 A3 \) have \( (\TargetTerm ([S']), Q') \in \Rel \)
unfolding \trans-def
by blast
with \( B4 A2 \) have \( ([S'], T) \in \TRel \)
by simp
with \( B3 \) show \( \exists T. ([S] \land\rightarrow\Target T \land ([S'], T) \in \TRel \)
by blast

next
fix \( S T \)
from \( A1 \) have \( (\SourceTerm S, \TargetTerm ([S])) \in \Rel \)
by simp
moreover assume \( [S] \land\rightarrow\Target T \)
hence \( \TargetTerm ([S]) \land\rightarrow\STCal\ Source\ Target\ (\TargetTerm T) \)
by (simp add: \STCal-step(2))
ultimately obtain \( P' \) where \( C1: \SourceTerm S \land\rightarrow\STCal\ Source\ Target\ P' \)
and \( C2: (P', \TargetTerm T) \in \Rel \)

using \( A4 \)
by blast
from \( C1 \) obtain \( S' \) where \( C3: S \land\rightarrow\Source S' \) and \( C4: S' \in S P' \)
by (auto simp add: \STCal-step(1))
from \( A1 C4 \) have \( (\TargetTerm ([S']), P') \in \Rel \)
by simp
from \( A3 \) this \( C2 \) have \( (\TargetTerm ([S']), \TargetTerm T) \in \Rel \)
unfolding \trans-def
by blast
with \( A2 \) have \( ([S'], T) \in \TRel \)
by simp
with \( C3 \) show \( \exists S'. S \land\rightarrow\Source S' \land ([S'], T) \in \TRel \)
by blast

qed
moreover have strong-reduction-bisimulation \( \TRel\ \Target \)

proof
auto
fix \( TP TQ TP' \)
assume \( (TP, TQ) \in \TRel \)
with \( A2 \) have \( (\TargetTerm TP, \TargetTerm TQ) \in \Rel \)
by simp
moreover assume \( TP \land\rightarrow\Target TP' \)
hence \( \TargetTerm TP \land\rightarrow\STCal\ Source\ Target\ (\TargetTerm TP') \)
by (simp add: \STCal-step(2))
ultimately obtain \( Q' \) where \( D1: \TargetTerm TQ \land\rightarrow\STCal\ Source\ Target\ Q' \)
and \( D2: (\TargetTerm TP', Q') \in \Rel \)

using \( A4 \)
by blast
from \( D1 \) obtain \( TQ' \) where \( D3: TQ \land\rightarrow\Target TQ' \) and \( D4: TQ' \in T Q' \)
by (auto simp add: \STCal-step(2))
from \( A2 D2 D4 \) have \( (TP', TQ') \in \TRel \)
by simp
with \( D3 \) show \( \exists TQ'. TQ \land\rightarrow\Target TQ' \land (TP', TQ') \in \TRel \)
by blast

next
fix \( TP TQ TQ' \)
assume \( (TP, TQ) \in \TRel \)
with \( A2 \) have \( (\TargetTerm TP, \TargetTerm TQ) \in \Rel \)
by simp
moredover assume \( TQ \mapsto \text{Target} \ TQ' \)
hence \( \text{TargetTerm} \ TQ \mapsto (\text{STCal Source Target}) (\text{TargetTerm} \ TQ') \)
by (simp add: \text{STCal-step}(2))
ultimately obtain \( P' \) where \( E1: \text{TargetTerm} \ TP \mapsto (\text{STCal Source Target}) P' \)
and \( E2: (P', \text{TargetTerm} \ TQ') \in \text{Rel} \)
using \( A4 \)
by blast
from \( E1 \) obtain \( TP' \) where \( E3: TP \mapsto \text{Target} TP' \)
and \( E4: TP' \in TP' \)
by simp
with \( E3 \) show \( \exists TP'. TP \mapsto \text{Target} TP' \land (TP', TQ') \in \text{TRel} \)
by blast
qed
ultimately
show \( \text{strongly-operational-corresponding TRel} \land \text{strong-reduction-bisimulation TRel} \text{Target} \)
by simp
qed

end

theory FullAbstraction
  imports SourceTargetRelation
begin

9 Full Abstraction

An encoding is fully abstract w.r.t. some source term relation \( SRel \) and some target term relation \( TRel \) if two source terms \( S1 \) and \( S2 \) form a pair \((S1, S2)\) in \( SRel \) iff their literal translations form a pair \((\text{enc} S1, \text{enc} S2)\) in \( TRel \).

abbreviation (\text{in encoding}) fully-abstract :: \( ('\text{procS} \times '\text{procS}) \Rightarrow ('\text{procT} \times '\text{procT}) \Rightarrow \text{bool} \)
where
\( \text{fully-abstract} \ SRel \ TRel \equiv \forall S1 S2. \ ([S1] = [S2] \rightarrow S1 = S2) \)

9.1 Trivial Full Abstraction Results

We start with some trivial full abstraction results. Each injective encoding is fully abstract w.r.t. to the identity relation on the source and the identity relation on the target.

lemma (in encoding) inj-encl-is-fully-abstract-wrt-identities:
assumes \( \text{injectivity} \equiv \forall S1 S2. \ [S1] = [S2] \rightarrow S1 = S2 \)
shows \( \text{fully-abstract} \ \{(S1, S2). S1 = S2\} \ (\{T1, T2\}. T1 = T2) \)
by (auto simp add: \text{injectivity})

Each encoding is fully abstract w.r.t. the empty relation on the source and the target.

lemma (in encoding) fully-abstract-wrt-empty-relation:
shows \( \text{fully-abstract} \ \{\} \ \{\} \)
by auto

Similarly, each encoding is fully abstract w.r.t. the all-relation on the source and the target.

lemma (in encoding) fully-abstract-wrt-all-relation:
shows \( \text{fully-abstract} \ \{(S1, S2). \text{True}\} \ \{(T1, T2). \text{True}\} \)
by auto

If the encoding is injective then for each source term relation \( \text{RelS} \) there exists a target term relation \( \text{RelT} \) such that the encoding is fully abstract w.r.t. \( \text{RelS} \) and \( \text{RelT} \).

lemma (in encoding) fully-abstract-wrt-source-relation:
If all source terms that are translated to the same target term are related by a trans source term relation \(\text{RelS}\), then there exists a target term relation \(\text{RelT}\) such that the encoding is fully abstract w.r.t. \(\text{RelS}\) and \(\text{RelT}\).

**lemma (in encoding) fully-abstract-wrt-trans-source-relation:**

*fixes* \(\text{RelS} : (\text{procS} \times \text{procS}) \text{ set}

*assumes* injectivity: \(\forall S1 S2. \ [S1] = [S2] \rightarrow S1 = S2\)

*shows* \(\exists \text{RelT}. \text{fully-abstract} \ \text{RelS} \ \text{RelT}\)

**proof** –

*define* \(\text{RelT}\) *where* \(\text{RelT} = \{(T1, T2). \ \exists S1 S2. (S1, S2) \in \text{RelS} \land T1 = [S1] \land T2 = [S2]\}\)

*with* injectivity *have* fully-abstract \(\text{RelS} \ \text{RelT}\)

*by* blast

**thus** \(\exists \text{RelT}. \text{fully-abstract} \ \text{RelS} \ \text{RelT}\)

*by* blast

**qed**

For every encoding and every target term relation \(\text{RelT}\) there exists a source term relation \(\text{RelS}\) such that the encoding is fully abstract w.r.t. \(\text{RelS}\) and \(\text{RelT}\).

**lemma (in encoding) fully-abstract-wrt-target-relation:**

*fixes* \(\text{RelS} : (\text{procS} \times \text{procS}) \text{ set}

*assumes* encRelS: \(\forall S1 S2. \ [S1] = [S2] \rightarrow (S1, S2) \in \text{RelS}\)

*shows* \(\exists \text{RelT}. \text{fully-abstract} \ (\text{RelS}^+) \ \text{RelT}\)

*using* encRelS trans-trancl[of RelS] fully-abstract-wrt-trans-source-relation[\textit{where} \text{RelS}=\text{RelS}^+]

*by* blast

**For every encoding and every target term relation \(\text{RelT}\) there exists a source term relation \(\text{RelS}\) such that the encoding is fully abstract w.r.t. \(\text{RelS}\) and \(\text{RelT}\).**

**lemma (in encoding) fully-abstract-wrt-source-relation:**
9.2 Fully Abstract Encodings

Thus, as long as we can choose one of the two relations, full abstraction is trivial. For fixed source and target relation terms encodings are not trivially fully abstract. For all encodings and relations SRel and TRel we can construct a relation on the disjunctive union of source and target terms, whose reduction to source terms is SRel and whose reduction to target terms is TRel. But full abstraction ensures that each trans relation that relates source terms and their literal translations in both directions includes SRel if it includes TRel restricted to translated source terms.

lemma (in encoding) full-abstraction-and-trans-relation-contains-SRel-impl-TRel:
  fixes Rel :: ('procS, 'procT) Proc × ('procS, 'procT) Proc set
  and SRel :: ('procS × 'procS) set
  and TRel :: ('procT × 'procT) set
  assumes fullAbs: fully-abstract SRel TRel
    and encR: ∀S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
    and srel: SRel = {(S1, S2). (SourceTerm S1, SourceTerm S2) ∈ Rel}
    and trel: trans (Rel ∪ {(P, Q). ∃S. [S] ∈ T P ∧ S ∈ S Q})
  shows ∀S1 S2. ([S1], [S2]) ∈ TRel ←→ (TargetTerm ([S1]), TargetTerm ([S2])) ∈ Rel
proof auto
  fix S1 S2
define Rel' where Rel' = Rel ∪ {(P, Q). ∃S. [S] ∈ T P ∧ S ∈ S Q}
hence (TargetTerm ([S1]), SourceTerm S1) ∈ Rel'
    by simp
  moreover assume ([S1], [S2]) ∈ TRel
  with fullAbs have (SourceTerm S1, SourceTerm S2) ∈ SRel
    by simp
  with srel have (SourceTerm S1, SourceTerm S2) ∈ Rel'
    by simp
  moreover from encR have (SourceTerm S2, TargetTerm ([S2])) ∈ Rel'
    by simp
  ultimately show (TargetTerm ([S1]), TargetTerm ([S2])) ∈ Rel
  using trans Rel'-def
  unfolding trans-def
  by blast
next
  fix S1 S2
define Rel' where Rel' = Rel ∪ {(P, Q). ∃S. [S] ∈ T P ∧ S ∈ S Q}
from encR have (SourceTerm S1, TargetTerm ([S1])) ∈ Rel'
  by simp
  moreover assume (TargetTerm ([S1]), TargetTerm ([S2])) ∈ Rel
  with Rel'-def have (TargetTerm ([S1]), TargetTerm ([S2])) ∈ Rel'
    by simp
  moreover from Rel'-def have (TargetTerm ([S2]), SourceTerm S2) ∈ Rel'
    by simp
  ultimately have (SourceTerm S1, SourceTerm S2) ∈ Rel
  using trans Rel'-def
  unfolding trans-def
  by blast
with srel have (S1, S2) ∈ SRel
  by simp

qed
with fullAbs show \([S_1], [S_2]\) \(\in\) TRel
by simp
qed

**lemma (in encoding)** full-abstraction-and-trans-relation-contains-TRel-impl-SRel:

**fixes** Rel \(\::\) \((\text{procS}, \text{procT})\) Proc \(\times\) \((\text{procS}, \text{procT})\) Proc \(\) set
and SRel \(\::\) \((\text{procS} \times \text{procS})\) set
and TRel \(\::\) \((\text{procT} \times \text{procT})\) set

**assumes** fullAbs : fully-abstract SRel TRel
and encR : \(\forall S.\) (SourceTerm S, TargetTerm ([S])) \(\in\) Rel
and trel : \(\forall S_1 S_2.\) ([S_1], [S_2]) \(\in\) TRel \(\longleftrightarrow\) (TargetTerm ([S_1]), TargetTerm ([S_2])) \(\in\) Rel

**shows** SRel = \{([S_1], [S_2]). (SourceTerm S_1, SourceTerm S_2) \(\in\) Rel\}

**proof**

**fix** S1 S2
**define** Rel' where Rel' = Rel \(\cup\) \{\((P, Q), \exists S.\) \([S] \in T P \land S \in S Q)\}
**from** encR Rel'-def have (SourceTerm S_1, TargetTerm ([S_1])) \(\in\) Rel'
by simp

moreover assume \((S_1, S_2)\) \(\in\) SRel
with fullAbs have \([S_1], [S_2]\) \(\in\) TRel
by simp

with trel Rel'-def have (TargetTerm ([S_1]), TargetTerm ([S_2])) \(\in\) Rel'
by simp

moreover from Rel'-def have (SourceTerm S_1, SourceTerm S_2) \(\in\) Rel'
by simp

ultimately show (SourceTerm S_1, SourceTerm S_2) \(\in\) Rel
using trans Rel'-def
unfolding trans-def
by blast

next

**fix** S1 S2
**define** Rel' where Rel' = Rel \(\cup\) \{\((P, Q), \exists S.\) \([S] \in T P \land S \in S Q)\}
**hence** (TargetTerm ([S_1]), SourceTerm S_1) \(\in\) Rel'
by simp

moreover assume \((SourceTerm S_1, SourceTerm S_2)\) \(\in\) Rel
with Rel'-def have \((SourceTerm S_1, SourceTerm S_2)\) \(\in\) Rel'
by simp

moreover from encR Rel'-def have (SourceTerm S_2, TargetTerm ([S_2])) \(\in\) Rel'
by simp

ultimately have (TargetTerm ([S_1]), TargetTerm ([S_2])) \(\in\) Rel
using trans Rel'-def
unfolding trans-def
by blast

with trel have \([S_1], [S_2]\) \(\in\) TRel
by simp

with fullAbs show \((S_1, S_2)\) \(\in\) SRel
by simp

qed

**lemma (in encoding)** full-abstraction-impl-trans-relation-contains-SRel-iff-TRel:

**fixes** Rel \(\::\) \((\text{procS}, \text{procT})\) Proc \(\times\) \((\text{procS}, \text{procT})\) Proc \(\) set
and SRel \(\::\) \((\text{procS} \times \text{procS})\) set
and TRel \(\::\) \((\text{procT} \times \text{procT})\) set

**assumes** fullAbs : fully-abstract SRel TRel
and encR : \(\forall S.\) (SourceTerm S, TargetTerm ([S])) \(\in\) Rel
and trans: \(\forall S_1 S_2.\) \(((S_1, [S_2]) \in TRel \longleftrightarrow (TargetTerm ([S_1]), TargetTerm ([S_2])) \in Rel) \leftrightarrow (SRel = \{([S_1], [S_2]). (SourceTerm S_1, SourceTerm S_2) \(\in\) Rel\})

**proof**

**assume** \((S_1, S_2)\) \(\in\) TRel = \((TargetTerm ([S_1]), TargetTerm ([S_2]))\) \(\in\) Rel
**thus** SRel = \{([S_1], [S_2]). (SourceTerm S_1, SourceTerm S_2) \(\in\) Rel\}

204
lemma (in encoding) full-abstraction-and-trans-relation-contains-SRel-iff-TRel-impl-TRel-encRL:

fixes Rel :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) set
and SRel :: ('procS × 'procS) set
and TRel :: ('procT × 'procT) set

assumes fullAbs: fully-abstract SRel TRel
and encR: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
and encL: ∀ S. (TargetTerm ([S]), SourceTerm S) ∈ Rel
and trans: trans Rel

shows (∀ S1 S2. ([S1], [S2]) ∈ TRel ↔ (TargetTerm ([S1]), TargetTerm ([S2])) ∈ Rel)
       → (SRel = {([S1], [S2]). (SourceTerm S1, SourceTerm S2) ∈ Rel})

proof
  from encL have Rel ∪ {(P, Q). ∃ S. [S] ∈ T P ∧ S ∈ S Q} = Rel
     by auto
  with fullAbs encR trans show ?thesis
     using full-abstraction-impl-trans-relation-contains-SRel-iff-TRel[where Rel=Rel
                      and SRel=SRel and TRel=TRel]
        by simp

qed

Full abstraction ensures that SRel and TRel satisfy the same basic properties that can be defined on their pairs. In particular: (1) SRel is refl if TRel reduced to translated source terms is refl (2) if the encoding is surjective then SRel is refl if TRel is refl (3) SRel is sym if TRel reduced to translated source terms is sym (4) SRel is trans iff TRel reduced to translated source terms is trans

lemma (in encoding) full-abstraction-impl-SRel-iff-TRel-is-refl:

fixes Rel :: (('procS × 'procS) set
and TRel :: ('procT × 'procT) set

assumes fullAbs: fully-abstract SRel TRel

shows refl SRel ↔ (∀ S. ([S], [S]) ∈ TRel)
  unfolding refl-on-def
  by (simp add: fullAbs)

lemma (in encoding) full-abstraction-and-surjectivity-impl-SRel-iff-TRel-is-refl:

fixes SRel :: ('procS × 'procS) set
and TRel :: ('procT × 'procT) set

assumes fullAbs: fully-abstract SRel TRel
and surj: ∀ T. ∃ S. T = [S]

shows refl SRel ↔ refl TRel

proof
  assume reflS: refl SRel
  show refl TRel
    unfolding refl-on-def
    proof auto
      fix T
      from surj obtain S where T = [S]
          by blast
      moreover from reflS have (S, S) ∈ SRel
        unfolding refl-on-def
          by simp
      with fullAbs have ([S], [S]) ∈ TRel

by simp
ultimately show \((T, T) \in TRel\)
by simp
qed

next
assume refl \(TRel\)
with fullAbs show refl \(SRel\)
  unfolding refl-on-def
by simp
qed

lemma (in encoding) full-abstraction-impl-SRel-iff-TRel-is-sym:
  fixes \(SRel\) :: \('\text{proc} S \times \text{proc} S\) set
  and \(TRel\) :: \('\text{proc} T \times \text{proc} T\) set
  assumes fullAbs: fully-abstract \(SRel\) \(TRel\)
  shows sym \(SRel\) \(\iff\) sym \{(\(T1, T2\)) \. \(\exists S1 S2. T1 = [S1] \land T2 = [S2] \land (T1, T2) \in TRel\}\)
  unfolding sym-def
  by (simp add: fullAbs, blast)

lemma (in encoding) full-abstraction-and-surjectivity-impl-SRel-iff-TRel-is-sym:
  fixes \(SRel\) :: \('\text{proc} S \times \text{proc} S\) set
  and \(TRel\) :: \('\text{proc} T \times \text{proc} T\) set
  assumes fullAbs: fully-abstract \(SRel\) \(TRel\)
  shows surj \(SRel\) \(\iff\) surj \(\\forall T. \exists S. T = [S]\)
  using fullAbs surj
  full-abstraction-impl-SRel-iff-TRel-is-sym[where \(SRel=SRel\) and \(TRel=TRel\]n
  by auto

lemma (in encoding) full-abstraction-and-surjectivity-impl-SRel-iff-TRel-is-trans:
  fixes \(SRel\) :: \('\text{proc} S \times \text{proc} S\) set
  and \(TRel\) :: \('\text{proc} T \times \text{proc} T\) set
  assumes fullAbs: fully-abstract \(SRel\) \(TRel\)
  shows trans \(SRel\) \(\iff\) trans \{(\(T1, T2\)) \. \(\exists S1 S2. T1 = [S1] \land T2 = [S2] \land (T1, T2) \in TRel\}\)
  unfolding trans-def
  by (simp add: fullAbs, blast)

Similarly, a fully abstract encoding that respects a predicate ensures the this predicate is preserved, reflected, or respected by \(SRel\) if it is preserved, reflected, or respected by \(TRel\).

lemma (in encoding) full-abstraction-and-enc-respects-pred-impl-SRel-iff-TRel-preserve:
  fixes \(SRel\) :: \('\text{proc} S \times \text{proc} S\) set
  and \(TRel\) :: \('\text{proc} T \times \text{proc} T\) set
  and \(Pred\) :: \('\text{proc}, '\text{proc}\) \(Pred \Rightarrow \text{bool}\)
  assumes fullAbs: fully-abstract \(SRel\) \(TRel\)
  and encP: enc-respects-pred \(Pred\)
  shows rel-preserves-pred \{(P, Q) \. \exists SP SQ. SP \in S P \land SQ \in S Q \land (SP, SQ) \in SRel\} \(Pred\)
  \(\iff\) rel-preserves-pred \{(P, Q) \. \exists SP SQ. [SP] \in T P \land [SQ] \in T Q \land ([SP], [SQ]) \in TRel\} \(Pred\)
proof
assume presS: rel-preserves-pred \{(P, Q) \. \exists SP SQ. SP \in S P \land SQ \in S Q \land (SP, SQ) \in SRel\} \(Pred\)
show rel-preserves-pred \{(P, Q) \. \exists SP SQ. [SP] \in T P \land [SQ] \in T Q \land ([SP], [SQ]) \in TRel\} \(Pred\)
proof clarify
fix SP SQ
assume Pred (TargetTerm ([SP]))
with encP have Pred (SourceTerm SP)
  by simp
moreover assume ([SP], [SQ]) ∈ TRel
with fullAbs have (SP, SQ) ∈ SRel
  by simp
ultimately have Pred (SourceTerm SQ)
  using presS
  by blast
with encP show Pred (TargetTerm ([SQ]))
  by simp
qed

next
assume presT:
rel-preserves-pred {(P, Q). ∃ SP SQ. [SP] ∈ T P ∧ [SQ] ∈ T Q ∧ ([SP], [SQ]) ∈ TRel} Pred
show rel-preserves-pred {(P, Q). ∃ SP SQ. SP ∈ S P ∧ SQ ∈ S Q ∧ (SP, SQ) ∈ SRel} Pred
proof clarify
fix SP SQ
assume Pred (SourceTerm SP)
with encP have Pred (TargetTerm ([SP]))
  by simp
moreover assume (SP, SQ) ∈ SRel
with fullAbs have ([SP], [SQ]) ∈ TRel
  by simp
ultimately have Pred (TargetTerm ([SQ]))
  using presT
  by blast
with encP show Pred (SourceTerm SQ)
  by simp
qed

lemma (in encoding) full-abstraction-and-enc-respects-binary-pred-impl-SRel-iff-TRel-preserve:
fixes SRel :: ('procS × 'procS) set
  and TRel :: ('procT × 'procT) set
  and Pred :: ('procS, procT) Proc ⇒ bool
assumes fullAbs: fully-abstract SRel TRel
  and encP: enc-respects-binary-pred Pred
shows rel-preserves-binary-pred {(P, Q). ∃ SP SQ. SP ∈ S P ∧ SQ ∈ S Q ∧ (SP, SQ) ∈ SRel} Pred
  ⇔ rel-preserves-binary-pred
{(P, Q). ∃ SP SQ. [SP] ∈ T P ∧ [SQ] ∈ T Q ∧ ([SP], [SQ]) ∈ TRel} Pred
proof clarify
fix x SP SQ
assume Pred (TargetTerm ([SP])) x
with encP have Pred (SourceTerm SP) x
  by simp
moreover assume ([SP], [SQ]) ∈ TRel
with fullAbs have (SP, SQ) ∈ SRel
  by simp
ultimately have Pred (SourceTerm SQ) x
  using presS
  by blast
with encP show Pred (TargetTerm ([SQ])) x
  by simp

qed
next

assume presT:
rel-preserves-binary-pred \{(P, Q). \exists SP SQ. [SP] \in T P \land [SQ] \in T Q \land ([SP], [SQ]) \in TRel\} Pred

show rel-preserves-binary-pred \{(P, Q). \exists SP SQ. SP \in S P \land SQ \in S Q \land (SP, SQ) \in SRel\} Pred

proof clarify

fix x SP SQ
assume Pred (SourceTerm SP) x
with encP have Pred (TargetTerm ([SP])) x
  by simp
moreover assume (SP, SQ) \in SRel
with fullAbs have ([SP], [SQ]) \in TRel
  by simp
ultimately have Pred (TargetTerm ([SQ])) x
  using presT
  by blast
with encP show Pred (SourceTerm SQ) x
  by simp
qed

qed


lemma (in encoding) full-abstraction-and-enc-respects-pred-impl-SRel-iff-TRel-reflects:

fixes SRel :: ('procS \times 'procS) set
  and TRel :: ('procT \times 'procT) set
  and Pred :: ('procS, 'procT) Proc \Rightarrow bool
assumes fullAbs: fully-abstract SRel TRel
  and encP: enc-respects-pred Pred
shows rel-reflects-pred \{(P, Q). \exists SP SQ. SP \in S P \land SQ \in S Q \land (SP, SQ) \in SRel\} Pred
\iff\ rel-reflects-pred \{(P, Q). \exists SP SQ. [SP] \in T P \land [SQ] \in T Q \land ([SP], [SQ]) \in TRel\} Pred

proof clarify

fix SP SQ
assume Pred (TargetTerm ([SQ]))
with encP have Pred (SourceTerm SQ)
  by simp
moreover assume ([SP], [SQ]) \in TRel
with fullAbs have (SP, SQ) \in SRel
  by simp
ultimately have Pred (SourceTerm SP)
  using refS
  by blast
with encP show Pred (TargetTerm ([SP]))
  by simp
qed


next

assume refT:
rel-reflects-pred \{(P, Q). \exists SP SQ. [SP] \in T P \land [SQ] \in T Q \land ([SP], [SQ]) \in TRel\} Pred

show rel-reflects-pred \{(P, Q). \exists SP SQ. SP \in S P \land SQ \in S Q \land (SP, SQ) \in SRel\} Pred

proof clarify

fix SP SQ
assume Pred (SourceTerm SQ)
with encP have Pred (TargetTerm ([SQ]))
  by simp
moreover assume (SP, SQ) \in SRel
with fullAbs have ([SP], [SQ]) \in TRel
  by simp
ultimately have Pred (TargetTerm ([SP]))
  using refT
  by blast
with \( encP \) show \( \text{Pred} \) \((\text{SourceTerm} \ SP)\)
by simp

qed

qed

lemma (in encoding) full-abstraction-and-enc-respects-binary-pred-impl-SRel-iff-TRel-reflects:

fixes \( SRel \) :: \((\text{procS} \times \text{procS})\) set
and \( TRel \) :: \((\text{procT} \times \text{procT})\) set
and \( \text{Pred} \) :: \((\text{procS}, \text{procT})\) \( \text{Proc} \Rightarrow 'b \Rightarrow \text{bool} \)
assumes \( \text{fullAbs} : \text{fully-abstract} \ SRel \ TRel \)
and \( \text{encP} : \text{enc-respects-binary-pred} \text{Pred} \)
shows \( \text{rel-reflects-binary-pred} \ ((P, Q). \exists SP \ SQ. SP \in S \ P \land SQ \in S \ Q \land (SP, SQ) \in SRel) \text{Pred} \)
\iff \( \text{rel-reflects-binary-pred} \ ((P, Q). \exists SP \ SQ. \exists T P \land T Q \land ([SP], [SQ]) \in TRel) \text{Pred} \)

proof

assume reflS:
rel-reflects-binary-pred \((P, Q). \exists SP \ SQ. SP \in S \ P \land SQ \in S \ Q \land (SP, SQ) \in SRel) \text{Pred} \)
show rel-reflects-binary-pred 
\((P, Q). \exists SP \ SQ. \exists T P \land T Q \land ([SP], [SQ]) \in TRel) \text{Pred} \)

proof clarify
fix \( x \) \( SP \ SQ \)
assume \( \text{Pred} \) \((\text{TargetTerm} ([SQ])) \ x \)
with \( \text{encP} \) have \( \text{Pred} \) \((\text{SourceTerm} \ SQ) \ x \)
by simp

moreover assume \([SP], [SQ]) \in TRel \)
with \( \text{fullAbs} \) have \((SP, SQ) \in SRel \)
by simp

ultimately have \( \text{Pred} \) \((\text{SourceTerm} \ SP) \ x \)
using reflS
by blast

with \( \text{encP} \) show \( \text{Pred} \) \((\text{TargetTerm} ([SP])) \ x \)
by simp

qed

next

assume reflT:
rel-reflects-binary-pred \((P, Q). \exists SP \ SQ. \exists T P \land T Q \land ([SP], [SQ]) \in TRel) \text{Pred} \)
show rel-reflects-binary-pred \((P, Q). \exists SP \ SQ. SP \in S \ P \land SQ \in S \ Q \land (SP, SQ) \in SRel) \text{Pred} \)

proof clarify
fix \( x \) \( SP \ SQ \)
assume \( \text{Pred} \) \((\text{SourceTerm} \ SQ) \ x \)
with \( \text{encP} \) have \( \text{Pred} \) \((\text{TargetTerm} ([SQ])) \ x \)
by simp

moreover assume \((SP, SQ) \in SRel \)
with \( \text{fullAbs} \) have \([SP], [SQ]) \in TRel \)
by simp

ultimately have \( \text{Pred} \) \((\text{TargetTerm} ([SP])) \ x \)
using reflT
by blast

with \( \text{encP} \) show \( \text{Pred} \) \((\text{SourceTerm} \ SP) \ x \)
by simp

qed

lemma (in encoding) full-abstraction-and-enc-respects-pred-impl-SRel-iff-TRel-respects:

fixes \( SRel \) :: \((\text{procS} \times \text{procS})\) set
and \( TRel \) :: \((\text{procT} \times \text{procT})\) set
and \( \text{Pred} \) :: \((\text{procS}, \text{procT})\) \( \text{Proc} \Rightarrow 'b \Rightarrow \text{bool} \)
assumes \( \text{fullAbs} : \text{fully-abstract} \ SRel \ TRel \)
and \( \text{encP} : \text{enc-respects-pred} \text{Pred} \)
shows \( \text{rel-respects-pred} \ ((P, Q). \exists SP \ SQ. SP \in S \ P \land SQ \in S \ Q \land (SP, SQ) \in SRel) \text{Pred} \)
\iff \( \text{rel-respects-pred} \ ((P, Q). \exists SP \ SQ. \exists T P \land T Q \land ([SP], [SQ]) \in TRel) \text{Pred} \)
using assms full-abstraction-and-enc-respects-pred-impl-SRel-iff-TRel-preserves[where SRel=SRel and TRel=TRel and Pred=Pred]
full-abstraction-and-enc-respects-pred-impl-SRel-iff-TRel-reflects[where SRel=SRel and TRel=TRel and Pred=Pred]

by auto

lemma (in encoding) full-abstraction-and-enc-respects-binary-impl-SRel-iff-TRel-respects:
fixes SRel :: ('procS × 'procS) set
and TRel :: ('procT × 'procT) set
and Pred :: ('procS, 'procT) Proc ⇒ 'b ⇒ bool
assumes fullAbs: fully-abstract SRel TRel
and encP: enc-respects-binary-pred Pred
shows rel-respects-binary-pred {{(P, Q). ∃SP SQ. SP ∈ S P ∧ SQ ∈ S Q ∧ (SP, SQ) ∈ SRel}} Pred
←→ rel-respects-binary-pred {{(P, Q). ∃SP SQ. [SP] ∈ T P ∧ [SQ] ∈ T Q ∧ ([SP], [SQ]) ∈ TRel}} Pred
using assms full-abstraction-and-enc-respects-binary-impl-SRel-iff-TRel-preserves[where SRel=SRel and TRel=TRel and Pred=Pred]
full-abstraction-and-enc-respects-binary-impl-SRel-iff-TRel-reflects[where SRel=SRel and TRel=TRel and Pred=Pred]
by auto

9.3 Full Abstraction w.r.t. Preorders
If there however exists a trans relation Rel that relates source terms and their literal translations in both directions, then the encoding is fully abstract with respect to the reduction of Rel to source terms and the reduction of Rel to target terms.

lemma (in encoding) trans-source-target-relation-impl-full-abstraction:
fixes Rel :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) set
assumes enc: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
∧ (TargetTerm ([S]), SourceTerm S) ∈ Rel
and trans: trans Rel
shows fully-abstract {s1, s2}. (SourceTerm s1, SourceTerm s2) ∈ Rel
{{T1, T2}. (TargetTerm T1, TargetTerm T2) ∈ Rel}
proof auto
fix s1 s2
assume (SourceTerm s1, SourceTerm s2) ∈ Rel
with enc trans show (TargetTerm ([s1]), TargetTerm ([s2])) ∈ Rel
unfolding trans-def
by blast
next
fix s1 s2
assume (TargetTerm ([s1]), TargetTerm ([s2])) ∈ Rel
with enc trans show (SourceTerm s1, SourceTerm s2) ∈ Rel
unfolding trans-def
by blast
qed

lemma (in encoding) source-target-relation-impl-full-abstraction-wrt-trans-closures:
fixes Rel :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) set
assumes enc: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
∧ (TargetTerm ([S]), SourceTerm S) ∈ Rel
shows fully-abstract {s1, s2}. (SourceTerm s1, SourceTerm s2) ∈ Rel⁺
{{T1, T2}. (TargetTerm T1, TargetTerm T2) ∈ Rel⁺}
proof auto
fix s1 s2
from enc have (TargetTerm ([s1]), SourceTerm s1) ∈ Rel⁺
by blast
moreover assume (SourceTerm s1, SourceTerm s2) ∈ Rel⁺
ultimately have (TargetTerm ([s1]), SourceTerm s2) ∈ Rel⁺
by simp

210
moreover from \textit{enc} have \((\textit{SourceTerm} \ S2, \textit{TargetTerm} \ ([S2])) \in \textit{Rel}^+


by \textit{blast}

ultimately show \((\textit{TargetTerm} \ ([S1]), \textit{TargetTerm} \ ([S2])) \in \textit{Rel}^+


by \textit{simp}

next

fix \ S1 \ S2

from \textit{enc} have \((\textit{SourceTerm} \ S1, \textit{TargetTerm} \ ([S1])) \in \textit{Rel}^+


by \textit{blast}

moreover assume \((\textit{TargetTerm} \ ([S1]), \textit{TargetTerm} \ ([S2])) \in \textit{Rel}^+


ultimately have \((\textit{SourceTerm} \ S1, \textit{TargetTerm} \ ([S2])) \in \textit{Rel}^+


by \textit{simp}

moreover from \textit{enc} have \((\textit{TargetTerm} \ ([S2]), \textit{SourceTerm} \ S2) \in \textit{Rel}^+


by \textit{blast}

ultimately show \((\textit{SourceTerm} \ S1, \textit{SourceTerm} \ S2) \in \textit{Rel}^+


by \textit{simp}

qed

\textbf{lemma (in encoding) quasi-trans-source-target-relation-impl-full-abstraction:}

\textbf{fixes} \ \textit{Rel} \ :: \ (\textit{\'procS, \'procT \ Proc} \times \textit{\'procS, \'procT \ Proc}) \set

\textbf{and} \ \textit{SRel} \ :: \ (\textit{\'procS \times \'procS}) \set

\textbf{and} \ \textit{TRel} \ :: \ (\textit{\'procT \times \'procT}) \set

\textbf{assumes enc:} \ \forall \ S, \ (\textit{SourceTerm} \ S, \textit{TargetTerm} \ ([S])) \in \textit{Rel}

\land \ (\textit{TargetTerm} \ ([S]), \textit{SourceTerm} \ S) \in \textit{Rel}

\textbf{and} \ \textit{srel:} \ \textit{SRel} = \{(S1, S2). (\textit{SourceTerm} S1, \textit{SourceTerm} S2) \in \textit{Rel}\}

\textbf{and} \ \textit{trel:} \ \textit{TRel} = \{(T1, T2). (\textit{TargetTerm} T1, \textit{TargetTerm} T2) \in \textit{Rel}\}

\textbf{and} \ \textit{trans:} \ \forall P Q R. (P, Q) \in \textit{Rel} \land (Q, R) \in \textit{Rel} \land ((P \in \textit{ProcS} \land Q \in \textit{ProcT})

\land (P \in \textit{ProcT} \land Q \in \textit{ProcS})) \rightarrow (P, R) \in \textit{Rel}

\textbf{shows} \ \textit{fully-abstract SRel TRel}

\textbf{proof auto}

fix \ S1 \ S2

from \textit{enc} have \((\textit{TargetTerm} \ ([S1]), \textit{SourceTerm} \ S1) \in \textit{Rel}


by \textit{simp}

moreover assume \(S1, S2) \in \textit{SRel}

with \textit{srel} have \((\textit{SourceTerm} \ S1, \textit{SourceTerm} \ S2) \in \textit{Rel}


by \textit{simp}

ultimately have \((\textit{TargetTerm} \ ([S1]), \textit{SourceTerm} \ S2) \in \textit{Rel}


using \textit{trans}

by \textit{blast}

moreover from \textit{enc} have \((\textit{SourceTerm} \ S2, \textit{TargetTerm} \ ([S2])) \in \textit{Rel}


by \textit{simp}

ultimately have \((\textit{TargetTerm} \ ([S1]), \textit{TargetTerm} \ ([S2])) \in \textit{Rel}


using \textit{trans}

by \textit{blast}

with \textit{trel show} \([S1], [S2]) \in \textit{TRel}

by \textit{simp}

next

fix \ S1 \ S2

from \textit{enc} have \((\textit{SourceTerm} \ S1, \textit{TargetTerm} \ ([S1])) \in \textit{Rel}


by \textit{simp}

moreover assume \([S1], [S2]) \in \textit{TRel}

with \textit{trel} have \((\textit{TargetTerm} \ ([S1]), \textit{TargetTerm} \ ([S2])) \in \textit{Rel}


by \textit{simp}

ultimately have \((\textit{SourceTerm} \ S1, \textit{TargetTerm} \ ([S2])) \in \textit{Rel}


using \textit{trans}

by \textit{blast}

moreover from \textit{enc} have \((\textit{TargetTerm} \ ([S2]), \textit{SourceTerm} \ S2) \in \textit{Rel}


by \textit{simp}

ultimately have \((\textit{SourceTerm} \ S1, \textit{SourceTerm} \ S2) \in \textit{Rel}


using \textit{trans}

by \textit{blast}

with \textit{srel show} \((S1, S2) \in \textit{SRel}


211
If an encoding is fully abstract w.r.t. SRel and TRel, then we can conclude from a pair in indRelRTPO or indRelSTEQ on a pair in TRel and SRel.

**lemma (in encoding) full-abstraction-impl-indRelRSTPO-to-SRel-and-TRel:**

**fixes** SRel :: ('procS × 'procS) set  
**and** TRel :: ('procT × 'procT) set  
**and** P Q :: ('procS, 'procT) Proc  
**assumes** fullAbs: fully-abstract SRel TRel  
**and** rel: P ≤[\[\[\cdot\]\]\R<\SRel,\TRel>] Q  
**shows** ∀ SP SQ. SP ∈ S P ∧ SQ ∈ S Q → ([SP], [SQ]) ∈ TRel⁺  
**and** ∀ SP TQ. SP ∈ S P ∧ TQ ∈ T Q → ([SP], TQ) ∈ TRel⁺  
**proof –**

- **have** fullAbsT: ∀ S1 S2. (S1, S2) ∈ SRel⁺ → ([S1], [S2]) ∈ TRel⁺  
  **proof** clarify  
  - **fix** S1 S2  
    - **assume** (S1, S2) ∈ SRel⁺  
    - **thus** ([S1], [S2]) ∈ TRel⁺  
  **proof induct**  
  - **fix** S2  
    - **assume** (S1, S2) ∈ SRel  
      **with** fullAbs show ([S1], [S2]) ∈ TRel⁺  
      **by simp**  
  - **next**  
    - **case** (step S2 S3)  
      - **assume** ([S1], [S2]) ∈ TRel⁺  
      - **moreover assume** (S2, S3) ∈ SRel  
        **with** fullAbs have ([S2], [S3]) ∈ TRel⁺  
        **by simp**  
      - **ultimately show** ([S1], [S3]) ∈ TRel⁺  
        **by simp**  
  - **qed**  
  - **qed**  
  **with** show ∀ SP SQ. SP ∈ S P ∧ SQ ∈ S Q → ([SP], [SQ]) ∈ TRel⁺  
    **using** indRelRSTPO-to-SRel-and-TRel(1)[where SRel=SRel and TRel=TRel]  
    **by simp**  
  **show** ∀ SP TQ. SP ∈ S P ∧ TQ ∈ T Q → ([SP], TQ) ∈ TRel⁺  
  **proof** clarify  
  - **fix** SP TQ  
    - **assume** SP ∈ S P ∧ TQ ∈ T Q  
      **with** rel obtain S where A1: (SP, S) ∈ SRel⁺  
        **and** A2: ([S], TQ) ∈ TRel⁺  
      **using** indRelRSTPO-to-SRel-and-TRel(2)[where SRel=SRel and TRel=TRel]  
        **by blast**  
    - **from** A1 have SP = S ∨ (SP, S) ∈ SRel⁺  
      **using** rtrancl-eq-or-trancl[of SP S SRel]  
      **by blast**  
    - **with** fullAbsT have ([SP], [S]) ∈ TRel⁺  
      **by fast**  
    - **from** this A2 show ([SP], TQ) ∈ TRel⁺  
      **by simp**  
  - **qed**  
  **qed**  

**lemma (in encoding) full-abstraction-wrt-preorders-impl-indRelSTEQ-to-SRel-and-TRel:**

**fixes** SRel :: ('procS × 'procS) set  
**and** TRel :: ('procT × 'procT) set  
**and** P Q :: ('procS, 'procT) Proc  
**assumes** fA: fully-abstract SRel TRel
and \(\text{transT}: \text{trans TRel}\)
and \(\text{reflS}: \text{refl SRel}\)
and \(\text{rel}: \forall P \sim [\square SRel, TRel] \rightarrow Q\)
shows \(\forall SP SQ. SP \in S P \land SQ \in S Q \rightarrow (SP, SQ) \in SRel\)
and \(\forall SP SQ. SP \in S P \land SQ \in S Q \rightarrow ([SP], [SQ]) \in TRel\)
and \(\forall TP SQ. TP \in T P \land SQ \in S Q \rightarrow (TP, [SQ]) \in TRel\)
and \(\forall TP TQ. TP \in T P \land TQ \in T Q \rightarrow (TP, TQ) \in TRel\)
using rel
proof induct
case (encR S)
show \(\forall SP SQ. SP \in S \text{SourceTerm} S \land SQ \in S \text{TargetTerm} ([S]) \rightarrow (SP, SQ) \in SRel\)
and \(\forall SP SQ. SP \in S \text{SourceTerm} S \land SQ \in S \text{SourceTerm} ([S]) \rightarrow ([SP], [SQ]) \in TRel\)
and \(\forall TP SQ. TP \in T \text{SourceTerm} S \land SQ \in S \text{SourceTerm} ([S]) \rightarrow (TP, [SQ]) \in TRel\)
and \(\forall TP TQ. TP \in T \text{SourceTerm} S \land TQ \in T \text{SourceTerm} ([S]) \rightarrow (TP, TQ) \in TRel\)
by simp+
from reflS JA show \(\forall SP TQ. SP \in S \text{SourceTerm} S \land TQ \in T \text{TargetTerm} ([S]) \rightarrow ([SP], TQ) \in TRel\)
unfolding refl-on-def
by simp
next
case (encL S)
show \(\forall SP SQ. SP \in S \text{SourceTerm} S \land SQ \in S \text{SourceTerm} S \rightarrow (SP, SQ) \in SRel\)
and \(\forall SP SQ. SP \in S \text{SourceTerm} S \land SQ \in S \text{SourceTerm} S \rightarrow ([SP], [SQ]) \in TRel\)
and \(\forall TP SQ. TP \in T \text{SourceTerm} S \land SQ \in S \text{SourceTerm} S \rightarrow (TP, [SQ]) \in TRel\)
and \(\forall TP TQ. TP \in T \text{SourceTerm} S \land TQ \in T \text{SourceTerm} S \rightarrow (TP, TQ) \in TRel\)
by simp+
with reflS JA show \(\forall TP SQ. TP \in T \text{SourceTerm} ([S]) \land SQ \in S \text{SourceTerm} S \rightarrow (TP, [SQ]) \in TRel\)
unfolding refl-on-def
by simp
next
case (source S1 S2)
show \(\forall SP TQ. SP \in S \text{SourceTerm} S1 \land TQ \in T \text{SourceTerm} S2 \rightarrow ([SP], TQ) \in TRel\)
and \(\forall TP SQ. TP \in T \text{SourceTerm} S1 \land SQ \in S \text{SourceTerm} S2 \rightarrow (TP, [SQ]) \in TRel\)
and \(\forall TP TQ. TP \in T \text{SourceTerm} S1 \land TQ \in T \text{SourceTerm} S2 \rightarrow (TP, TQ) \in TRel\)
by simp+
assume \((S1, S2) \in SRel\)
thus \(\forall SP SQ. SP \in S \text{SourceTerm} S1 \land SQ \in S \text{SourceTerm} S2 \rightarrow (SP, SQ) \in SRel\)
by simp
with JA show \(\forall SP SQ. SP \in S \text{SourceTerm} S1 \land SQ \in S \text{SourceTerm} S2 \rightarrow ([SP], [SQ]) \in TRel\)
by simp
next
case (target T1 T2)
show \(\forall SP SQ. SP \in S \text{SourceTerm} T1 \land SQ \in S \text{SourceTerm} T2 \rightarrow ([SP], SQ) \in SRel\)
and \(\forall SP SQ. SP \in S \text{SourceTerm} T1 \land SQ \in S \text{SourceTerm} T2 \rightarrow ([SP], SQ) \in TRel\)
and \(\forall TP SQ. TP \in T \text{SourceTerm} T1 \land TQ \in T \text{SourceTerm} T2 \rightarrow ([SP], SQ) \in TRel\)
and \(\forall TP SQ. TP \in T \text{SourceTerm} T1 \land SQ \in S \text{SourceTerm} T2 \rightarrow (TP, [SQ]) \in TRel\)
by simp+
assume \((T1, T2) \in TRel\)
thus \(\forall TP TQ. TP \in T \text{SourceTerm} T1 \land TQ \in T \text{SourceTerm} T2 \rightarrow (TP, TQ) \in TRel\)
by simp
next
case (trans P Q R)
assume A1: \(\forall SP SQ. SP \in S P \land SQ \in S Q \rightarrow ([SP], [SQ]) \in TRel\)
and A2: \(\forall SP TQ. SP \in S P \land TQ \in T Q \rightarrow ([SP], TQ) \in TRel\)
and A3: \(\forall TP SQ. TP \in T P \land SQ \in S Q \rightarrow (TP, [SQ]) \in TRel\)
and A4: \(\forall TP TQ. TP \in T P \land TQ \in T Q \rightarrow (TP, TQ) \in TRel\)
and A5: \(\forall SQ SR. SQ \in S Q \land SR \in S R \rightarrow ([SQ], [SR]) \in TRel\)
and A6: \(\forall SQ TR. SQ \in S Q \land TR \in T R \rightarrow ([SQ], TR) \in TRel\)
and A7: \(\forall TQ SR. TQ \in T Q \land SR \in S R \rightarrow (TQ, [SR]) \in TRel\)
and A8: \(\forall TQ TR. TQ \in T Q \land TR \in T R \rightarrow (TQ, TR) \in TRel\)
show \(\forall SP SR. SP \in S P \land SR \in S R \rightarrow ([SP], [SR]) \in TRel\)
proof clarify
fix $SP$ $SR$
assume $A9$: $SP \in S$ $P$ and $A10$: $SR \in S$ $R$
show $([SP], [SR]) \in TRel$

proof (cases $Q$)
case (SourceTerm $SQ$)
assume $A11$: $SQ \in S$ $Q$
with $A1$ $A9$ have $([SP], [SQ]) \in TRel$
by blast
moreover from $A5$ $A10$ $A11$ have $([SQ], [SR]) \in TRel$
by blast
ultimately show $([SP], [SR]) \in TRel$
using $transT$
unfolding $trans-def$
by blast

next
case (TargetTerm $TQ$)
assume $A11$: $TQ \in T$ $Q$
with $A2$ $A9$ have $([SP], TQ) \in TRel$
by blast
moreover from $A7$ $A10$ $A11$ have $(TQ, [SR]) \in TRel$
by blast
ultimately show $([SP], [SR]) \in TRel$
using $transT$
unfolding $trans-def$
by blast
qed

qed

with $fa$ show $\forall SP$ $SR$. $SP \in S$ $P$ $\land$ $SR \in S$ $R$ $\rightarrow$ $(SP, SR) \in SRel$
by $simp$
show $\forall SP$ $TR$. $SP \in S$ $P$ $\land$ $TR \in T$ $R$ $\rightarrow$ $([SP], TR) \in TRel$

proof clarify
fix $SP$ $TR$
assume $A9$: $SP \in S$ $P$ and $A10$: $TR \in T$ $R$
show $([SP], TR) \in TRel$

proof (cases $Q$)
case (SourceTerm $SQ$)
assume $A11$: $SQ \in S$ $Q$
with $A1$ $A9$ have $([SP], [SQ]) \in TRel$
by blast
moreover from $A6$ $A10$ $A11$ have $([SQ], TR) \in TRel$
by blast
ultimately show $([SP], TR) \in TRel$
using $transT$
unfolding $trans-def$
by blast

next
case (TargetTerm $TQ$)
assume $A11$: $TQ \in T$ $Q$
with $A2$ $A9$ have $([SP], TQ) \in TRel$
by blast
moreover from $A8$ $A10$ $A11$ have $(TQ, TR) \in TRel$
by blast
ultimately show $([SP], TR) \in TRel$
using $transT$
unfolding $trans-def$
by blast
qed

qed

show $\forall TP$ $SR$. $TP \in T$ $P$ $\land$ $SR \in S$ $R$ $\rightarrow$ $(TP, [SR]) \in TRel$

proof clarify
If an encoding is fully abstract w.r.t. a preorder SRel on the source and a trans relation TRel on the target, then there exists a trans relation, namely indRelSTEQ, that relates source terms and their literal translations in both direction such that its reductions to source terms is SRel and its reduction...
to target terms is TRel.

**lemma (in encoding) full-abstraction-wrt-preorders-impl-trans-source-target-relation:**

fixes $SRel :: (\texttt{procS} \times \texttt{procS})$ set
and $TRel :: (\texttt{procT} \times \texttt{procT})$ set
assumes fullAbs: fully-abstract $SRel \ TRel$
and reflS: refl $SRel$
and transT: trans $TRel$
shows $\exists \text{Rel.} (\forall S \cdot (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel})$
\quad $\wedge SRel = \{(S1, S2), (\text{SourceTerm} S1, \text{SourceTerm} S2) \in \text{Rel}\}$
\quad $\wedge TRel = \{(T1, T2), (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel}\}$
\quad $\wedge \text{trans Rel}$

proof –
have $\forall S \cdot \text{SourceTerm} S \sim SRel, TRel > \text{TargetTerm} ([S])$
\quad $\wedge \text{TargetTerm} ([S]) \sim TRel > \text{SourceTerm} S$
using $\text{indRelSTEQ.encR[where SRel=SRel and TRel=TRel]}$
$\text{indRelSTEQ.encL[where SRel=SRel and TRel=TRel]}$ by blast
moreover have $SRel = \{(S1, S2), \text{SourceTerm} S1 \sim SRel, TRel > \text{SourceTerm} S2\}$
proof auto
fix $S1 \ S2$
assume $(S1, S2) \in SRel$
thus $\text{SourceTerm} S1 \sim SRel, TRel > \text{SourceTerm} S2$
by (rule indRelSTEQ.source[where SRel=SRel and TRel=TRel])
next
fix $S1 \ S2$
assume $\text{SourceTerm} S1 \sim SRel, TRel > \text{SourceTerm} S2$
with fullAbs reflT transT show $(S1, S2) \in SRel$
using full-abstraction-wrt-preorders-impl-indRelSTEQ-to-SRel-and-TRel(1)[where SRel=SRel
and TRel=TRel] by blast
moreover have $TRel = \{(T1, T2), \text{TargetTerm} T1 \sim TRel > \text{TargetTerm} T2\}$
proof auto
fix $T1 \ T2$
assume $(T1, T2) \in TRel$
thus $\text{TargetTerm} T1 \sim TRel > \text{TargetTerm} T2$
by (rule indRelSTEQ.target[where SRel=SRel and TRel=TRel])
next
fix $T1 \ T2$
assume $\text{TargetTerm} T1 \sim TRel > \text{TargetTerm} T2$
with fullAbs reflT transT show $(T1, T2) \in TRel$
using full-abstraction-wrt-preorders-impl-indRelSTEQ-to-SRel-and-TRel(5)[where SRel=SRel
and TRel=TRel] by blast
moreover have $\text{trans (indRelSTEQ SRel TRel)}$
using $\text{indRelSTEQ.trans[where SRel=SRel and TRel=TRel]}$
unfolding trans-def by blast
ultimately show thesis by blast
qed

Thus an encoding is fully abstract w.r.t. a preorder SRel on the source and a trans relation TRel on the
target iff there exists a trans relation that relates source terms and their literal translations in
both directions and whose reduction to source/target terms is SRel/TRel.

**theorem (in encoding) fully-abstract-wrt-preorders-iff-source-target-relation-is-trans:**

fixes $SRel :: (\texttt{procS} \times \texttt{procS})$ set
and $TRel :: (\texttt{procT} \times \texttt{procT})$ set
shows (fully-abstract SRel TRel ∧ refl SRel ∧ trans TRel) =
(∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
∧ (TargetTerm ([S]), SourceTerm S) ∈ Rel)
∧ SRel = {(S1, S2). (SourceTerm S1, SourceTerm S2) ∈ Rel}
∧ TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
∧ trans Rel)
proof (rule iffI)
assume fully-abstract SRel TRel ∧ refl SRel ∧ trans TRel
thus ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel ∧ (TargetTerm ([S]), SourceTerm S) ∈ Rel)
∧ SRel = {(S1, S2). (SourceTerm S1, SourceTerm S2) ∈ Rel}
∧ TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
∧ trans Rel
using full-abstraction-wrt-preorders-impl-trans-source-target-relation[where SRel=SRel
and TRel=TRel]
by blast
next
assume ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
∧ (TargetTerm ([S]), SourceTerm S) ∈ Rel)
∧ SRel = {(S1, S2). (SourceTerm S1, SourceTerm S2) ∈ Rel}
∧ TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
∧ trans Rel
from this obtain Rel
where A1: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel ∧ (TargetTerm ([S]), SourceTerm S) ∈ Rel
and A2: SRel = {(S1, S2). (SourceTerm S1, SourceTerm S2) ∈ Rel}
and A3: TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel} and A4: trans Rel
by blast
hence fully-abstract SRel TRel
using trans-source-target-relation-impl-full-abstraction[where Rel=Rel]
by blast
moreover have refl SRel
unfolding refl-on-def
proof auto
fix S
from A1 have (SourceTerm S, TargetTerm ([S])) ∈ Rel
by blast
moreover from A1 have (TargetTerm ([S]), SourceTerm S) ∈ Rel
by blast
ultimately have (SourceTerm S, SourceTerm S) ∈ Rel
using A4
unfolding trans-def
by blast
with A2 show (S, S) ∈ SRel
by blast
qed
moreover from A3 A4 have trans TRel
unfolding trans-def
by blast
ultimately show fully-abstract SRel TRel ∧ refl SRel ∧ trans TRel
by blast
qed

9.4 Full Abstraction w.r.t. Equivalences

If there exists a relation Rel that relates source terms and their literal translations and whose sym
closure is trans, then the encoding is fully abstract with respect to the reduction of the sym closure
of Rel to source/target terms.

lemma (in encoding) source-target-relation-with-trans-syncl-impl-full-abstraction:
fixes Rel :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) set
assumes enc: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
and trans: trans (syncl Rel)
shows fully-abstract \((S_1, S_2). (\text{SourceTerm } S_1, \text{SourceTerm } S_2) \in \text{symcl Rel}\) \((T_1, T_2). (\text{TargetTerm } T_1, \text{TargetTerm } T_2) \in \text{symcl Rel}\)

proof auto
fix \(S_1 \ S_2\)
from \(\text{enc}\) have \((\text{TargetTerm } ([S_1]), \text{SourceTerm } S_1) \in \text{symcl Rel}\)
by \((\text{simp add: symcl-def})\)
moreover assume \((\text{SourceTerm } S_1, \text{SourceTerm } S_2) \in \text{symcl Rel}\)
moreover from \(\text{enc}\) have \((\text{SourceTerm } S_2, \text{TargetTerm } ([S_2])) \in \text{symcl Rel}\)
by \((\text{simp add: symcl-def})\)
ultimately show \((\text{TargetTerm } ([S_1]), \text{TargetTerm } ([S_2])) \in \text{symcl Rel}\)
using trans
unfolding trans-def
by blast

next
fix \(S_1 \ S_2\)
from \(\text{enc}\) have \((\text{SourceTerm } S_1, \text{TargetTerm } ([S_1])) \in \text{symcl Rel}\)
by \((\text{simp add: symcl-def})\)
moreover assume \((\text{TargetTerm } ([S_1]), \text{TargetTerm } ([S_2])) \in \text{symcl Rel}\)
moreover from \(\text{enc}\) have \((\text{SourceTerm } S_2, \text{SourceTerm } S_2) \in \text{symcl Rel}\)
by \((\text{simp add: symcl-def})\)
ultimately show \((\text{SourceTerm } S_1, \text{SourceTerm } S_2) \in \text{symcl Rel}\)
using trans
unfolding trans-def
by blast

qed

If an encoding is fully abstract w.r.t. the equivalences \(\text{SRel}\) and \(\text{TRel}\), then there exists a preorder, namely \(\text{indRelRSTPO}\), that relates source terms and their literal translations such that its reductions to source terms is \(\text{SRel}\) and its reduction to target terms is \(\text{TRel}\).

lemma \(\text{in encoding}\) fully-abstract-wrt-equivalences-impl-symcl-source-target-relation-is-preorder:
fixes \(\text{SRel} :: (\text{'procS} \times \text{'procS}) \text{ set}\)
and \(\text{TRel} :: (\text{'procT} \times \text{'procT}) \text{ set}\)
assumes fullAbs: fully-abstract \(\text{SRel} \ \text{TRel}\)
and reflT: \(\text{refl TRel}\)
and symmT: \(\text{sym TRel}\)
and transT: \(\text{trans TRel}\)
shows \(\exists \text{Rel}. (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel})\)
\(\land \ \text{SRel} = \{(S_1, S_2). (\text{SourceTerm } S_1, \text{SourceTerm } S_2) \in \text{symcl Rel}\}\)
\(\land \ \text{TRel} = \{(T_1, T_2). (\text{TargetTerm } T_1, \text{TargetTerm } T_2) \in \text{symcl Rel}\}\)
\(\land \ \text{preorder (symcl Rel)}\)

proof –
from fullAbs reflT have reflS: \(\text{refl SRel}\)
unfolding refl-on-def
by auto
from fullAbs symmT have symmS: \(\text{sym SRel}\)
unfolding sym-def
by auto
from fullAbs transT have transS: \(\text{trans SRel}\)
unfolding trans-def
by blast
have \(\forall S. \text{SourceTerm } S \sqsubseteq [\cdot] R \sqsubset \text{SRel, TRel}> \text{TargetTerm } ([S])\)
using \(\text{indRelRSTPO.encR[where SRel=SRel and TRel=TRel]}\)
by blast
moreover
have \(\text{SRel} = \{(S_1, S_2). (\text{SourceTerm } S_1, \text{SourceTerm } S_2) \in \text{symcl (indRelRSTPO SRel TRel)}\}\)
proof auto
fix \(S_1 \ S_2\)
assume \((S_1, S_2) \in \text{SRel}\)
thus \((\text{SourceTerm } S_1, \text{SourceTerm } S_2) \in \text{symcl (indRelRSTPO SRel TRel)}\)
by \((\text{simp add: symcl-def indRelRSTPO.source[where SRel=SRel and TRel=TRel]}\)

218
next
fix \( S_1 \) \( S_2 \)
assume \((\text{SourceTerm } S_1, \text{SourceTerm } S_2) \in \text{symcl (indRelRSTPO } S\text{Rel } T\text{Rel})\)
moreover from \text{transS}
have \( \text{SourceTerm } S_1 \leq \{ \text{\| } R < S\text{Rel}, T\text{Rel}\} \quad \text{SourceTerm } S_2 \Rightarrow (S_1, S_2) \in S\text{Rel} \)
using \text{indRelRSTPO-to-SRel-and-TRel(1)}[\text{where } S\text{Rel}=S\text{Rel} \text{ and } T\text{Rel}=T\text{Rel}]
\text{by blast}
moreover from \text{symmS transS}
have \( \text{SourceTerm } S_2 \leq \{ \text{\| } R < S\text{Rel}, T\text{Rel}\} \quad \text{SourceTerm } S_1 \Rightarrow (S_1, S_2) \in S\text{Rel} \)
using \text{indRelRSTPO-to-SRel-and-TRel(4)}[\text{where } S\text{Rel}=S\text{Rel} \text{ and } T\text{Rel}=T\text{Rel}]
\text{unfolding sym-def}
\text{by blast}
ultimately show \((S_1, S_2) \in S\text{Rel}\)
by \text{(auto simp add: symcl-def)}
qed
moreover have \text{refl } (\text{symcl (indRelRSTPO } S\text{Rel } T\text{Rel}))
\text{unfolding refl-on-def}
proof \text{auto}
fix \( P \)
show \((P, P) \in \text{symcl (indRelRSTPO } S\text{Rel } T\text{Rel})\)
proof \text{(cases } P)\n\text{case } (\text{SourceTerm } SP)\n\text{assume } SP \in S \text{P} \n\text{with \text{reflS }} \text{show } ((P, P) \in \text{symcl (indRelRSTPO } S\text{Rel } T\text{Rel})\)
\text{unfolding refl-on-def}
\text{by (simp add: symcl-def indRelRSTPO.source)}
next\n\text{case } (\text{TargetTerm } TP)\n\text{assume } TP \in T \text{P} \n\text{with \text{reflT }} \text{show } ((P, P) \in \text{symcl (indRelRSTPO } S\text{Rel } T\text{Rel})\)
\text{unfolding refl-on-def}
\text{by (simp add: symcl-def indRelRSTPO.target)}
qed
qed
moreover have \text{trans } (\text{symcl (indRelRSTPO } S\text{Rel } T\text{Rel}))
proof

have \( \forall P Q R. P \leq [\cdot] R < S{\text{Rel}}, T{\text{Rel}} > Q \land R \leq [\cdot] R < S{\text{Rel}}, T{\text{Rel}} > Q \land (P, R) \notin (\text{indRelRSTPO} \ S{\text{Rel}} \ T{\text{Rel}}) \rightarrow Q \leq [\cdot] R < S{\text{Rel}}, T{\text{Rel}} > P \lor Q \leq [\cdot] R < S{\text{Rel}}, T{\text{Rel}} > R \)

proof clarify

fix \( P Q R \)

assume \( A1: P \leq [\cdot] R < S{\text{Rel}}, T{\text{Rel}} > Q \) and \( A2: R \leq [\cdot] R < S{\text{Rel}}, T{\text{Rel}} > Q \)

and \( A3: (P, R) \notin (\text{indRelRSTPO} \ S{\text{Rel}} \ T{\text{Rel}}) \) and \( A4: (Q, R) \notin (\text{indRelRSTPO} \ S{\text{Rel}} \ T{\text{Rel}}) \)

show \( Q \leq [\cdot] R < S{\text{Rel}}, T{\text{Rel}} > P \)

proof (cases \( P \))


case (\( \text{SourceTerm} \ SP \))

assume \( A5: SP \in S P \)

show \( Q \leq [\cdot] R < S{\text{Rel}}, T{\text{Rel}} > P \)

proof (cases \( Q \))


case (\( \text{SourceTerm} \ SQ \))

assume \( A6: SQ \in S Q \)

with \( \text{transS} \ A1 A5 \) have \( (SP, SQ) \in S{\text{Rel}} \)

using \( \text{indRelRSTPO-to-SRel-and-TRel}() \) [where \( S{\text{Rel}}=S{\text{Rel}} \) and \( T{\text{Rel}}=T{\text{Rel}} \)]

\( \text{transl-id}() \) [of \( S{\text{Rel}} \)]

by \( \text{blast} \)

with \( \text{symmS} \ A5 A6 \) show \( Q \leq [\cdot] R < S{\text{Rel}}, T{\text{Rel}} > P \)

unfolding \( \text{sym-def} \)

by \( \text{(simp add: indRelRSTPO.source)} \)

next


case (\( \text{TargetTerm} \ TQ \))

assume \( A6: TQ \in T Q \)

show \( Q \leq [\cdot] R < S{\text{Rel}}, T{\text{Rel}} > P \)

proof (cases \( R \))


case (\( \text{SourceTerm} \ SR \))

assume \( A7: SR \in S R \)

with \( \text{fullAbs} \ A2 A6 \) have \( ([SR], TQ) \in T{\text{Rel}}^* \)

using \( \text{full-abstraction-impl-indRelRSTPO-to-SRel-and-TRel}() \) [where \( S{\text{Rel}}=S{\text{Rel}} \) and \( T{\text{Rel}}=T{\text{Rel}} \)]

\( \text{transcl-id}() \) [of \( T{\text{Rel}}^* \)] \( \text{reflcl-of-refl-rel}() \) [of \( T{\text{Rel}} \)]

unfolding \( \text{trans-def} \)

by \( \text{blast} \)

with \( \text{transT} \ \text{refT} T \) have \( ([SP], TQ) \in T{\text{Rel}} \)

using \( \text{transl-id}() \) [of \( T{\text{Rel}}^* \)] \( \text{reflcl-of-refl-rel}() \) [of \( T{\text{Rel}} \)] \( \text{transcl-reflcl}() \) [of \( T{\text{Rel}} \)]

by \( \text{auto} \)

with \( \text{symmT} \) have \( (TQ, [SR]) \in T{\text{Rel}} \)

unfolding \( \text{sym-def} \)

by \( \text{simp} \)

moreover from \( \text{fullAbs} \ A1 A5 A6 \) have \( ([SP], TQ) \in T{\text{Rel}}^* \)

using \( \text{full-abstraction-impl-indRelRSTPO-to-SRel-and-TRel}() \) [where \( S{\text{Rel}}=S{\text{Rel}} \) and \( T{\text{Rel}}=T{\text{Rel}} \)]

unfolding \( \text{trans-def} \)

by \( \text{blast} \)

with \( \text{transT} \ \text{refT} T \) have \( ([SP], TQ) \in T{\text{Rel}} \)

using \( \text{transl-id}() \) [of \( T{\text{Rel}}^* \)] \( \text{reflcl-of-refl-rel}() \) [of \( T{\text{Rel}} \)] \( \text{transcl-reflcl}() \) [of \( T{\text{Rel}} \)]

by \( \text{auto} \)

ultimately have \( ([SP], [SR]) \in T{\text{Rel}} \)

using \( \text{transT} \)

unfolding \( \text{trans-def} \)

by \( \text{blast} \)

with \( \text{fullAbs} \) have \( (SP, SR) \in S{\text{Rel}} \)

by \( \text{simp} \)

with \( A3 A5 A7 \) show \( ?\text{thesis} \)

by \( \text{(simp add: indRelRSTPO.source)} \)

next


case (\( \text{TargetTerm} \ TR \))

assume \( A7: TR \in T R \)

with \( \text{transT} A2 A6 \) have \( (TR, TQ) \in T{\text{Rel}} \)
using `indRelRSTPO-to-SRel-and-TRel(4)`[where SRel=SRel and TRel=TRel]

trancl-id[of TRel]

by blast

with `symmT` have `(TQ, TR) ∈ TRel`

unfolding `sym-def`

by `simp`

with `A4 A6 A7` show `?thesis`

by `(simp add: indRelRSTPO.target)`

qed

qed

qed

moreover have ∀ P Q R. P ≲[\[\cdot\]] R <\[\[\cdot\]\] SRel, TRel> P ∧ P ≲[\[\cdot\]] R <\[\[\cdot\]\] SRel, TRel> R ∧ (Q, R) /∈ (indRelRSTPO SRel TRel)

→ Q ≲[\[\cdot\]] R <\[\[\cdot\]\] SRel, TRel> P ∨ R ≲[\[\cdot\]] R <\[\[\cdot\]\] SRel, TRel> P

proof clarify

fix P Q R

assume `A1: P ≲[\[\cdot\]] R <\[\[\cdot\]\] SRel, TRel> Q` and `A2: P ≲[\[\cdot\]] R <\[\[\cdot\]\] SRel, TRel> R` and `A3: (Q, R) /∈ (indRelRSTPO SRel TRel)` and `A4: (R, P) /∈ (indRelRSTPO SRel TRel)`

show `Q ≲[\[\cdot\]] R <\[\[\cdot\]\] SRel, TRel> P`

proof (cases `P`)

  case (SourceTerm `SP`)
  assume `A5: SP ∈ S P`
  show `Q ≲[\[\cdot\]] R <\[\[\cdot\]\] SRel, TRel> P`

  proof (cases `Q`)
    case (SourceTerm `SQ`)
    assume `A6: SQ ∈ S Q`
    with `transS A1 A5` have `(SP, SQ) ∈ SRel`

    using `indRelRSTPO-to-SRel-and-TRel(1)`[where SRel=SRel and TRel=TRel]

    trancl-id[of SRel]

    by blast

    with `symmS A5 A6` show `Q ≲[\[\cdot\]] R <\[\[\cdot\]\] SRel, TRel> P`

    unfolding `sym-def`

    by `(simp add: indRelRSTPO.source)`

  next

  case (TargetTerm `TQ`)
  assume `A6: TQ ∈ T Q`
  show `Q ≲[\[\cdot\]] R <\[\[\cdot\]\] SRel, TRel> P`

proof (cases `Q`)

  case (SourceTerm `SQ`)
  assume `A6: SQ ∈ S Q`
  with `transS A1 A5` have `(TQ, SQ) ∈ SRel`

  using `indRelRSTPO-to-SRel-and-TRel(4)`[where SRel=SRel and TRel=TRel]

  trancl-id[of TRel]

  by blast

  with `symmT` have `(SQ, TQ) ∈ TRel`

  unfolding `sym-def`

  by `(simp add: indRelRSTPO.target)`

qed

qed

next
proof (cases R)
  case (SourceTerm SR)
    assume A7: SR ∈ S R
    with transS A2 A5 have (SP, SR) ∈ SRel
      using indRelRSTPO-to-SRel-and-TRel(1)[where SRel=SRel and TRel=TRel]
      trancl-id[of SRel]
      by blast
    with symmS have (SR, SP) ∈ SRel
      unfolding sym-def
      by simp
    with A4 A5 A7 show ?thesis
      by (simp add: indRelRSTPO.source)
next
  case (TargetTerm TR)
    from fullAbs A1 A5 A6 have ([SP], TQ) ∈ TRel*
      using full-abstraction-impl-indRelRSTPO-to-SRel-and-TRel(2)[where SRel=SRel and TRel=TRel]
      unfolding trans-def
      by blast
    with transT reflT have ([SP], TQ) ∈ TRel
      using trancl-id[of TRel'=] refl-of-refl-rel[of TRel] trancl-refcl[of TRel]
      by auto
    with symmT have (TQ, [SP]) ∈ TRel
      unfolding sym-def
      by simp
    moreover assume A7: TR ∈ T R
    with fullAbs A2 A5 have ([SP], TR) ∈ TRel*
      using full-abstraction-impl-indRelRSTPO-to-SRel-and-TRel(2)[where SRel=SRel and TRel=TRel]
      unfolding trans-def
      by blast
    with transT reflT have ([SP], TR) ∈ TRel
      using trancl-id[of TRel'=] refl-of-refl-rel[of TRel] trancl-refcl[of TRel]
      by auto
    ultimately have (TQ, TR) ∈ TRel
      using transT
      unfolding trans-def
      by blast
    with A3 A6 A7 show ?thesis
      by (simp add: indRelRSTPO.target)
    qed
  qed
next
  case (TargetTerm TP)
    assume A5: TP ∈ T P
    show Q ≲ R< SRel,TRel> P
      proof (cases Q)
        case (SourceTerm SQ)
        assume SQ ∈ S Q
        with A1 A5 show ?thesis
          using indRelRSTPO-to-SRel-and-TRel(3)[where SRel=SRel and TRel=TRel]
          by blast
      next
        case (TargetTerm TQ)
        assume A6: TQ ∈ T Q
        with transT A1 A5 have (TP, TQ) ∈ TRel
          using indRelRSTPO-to-SRel-and-TRel(4)[where SRel=SRel and TRel=TRel]
          trancl-id[of TRel]
          by blast
        with symmT have (TQ, TP) ∈ TRel
          unfolding sym-def
by simp
with A5 A6 show \( Q \subseteq \{ \cdot \} R < S_{\text{Rel}} TRel > P \)
by (simp add: indRelRSTPO.target)
qed
qed qed
moreover from reflS reflT have refl (indRelRSTPO SRel TRel)
using indRelRSTPO-refl[where SRel=SRel and TRel=TRel]
by blast
moreover have trans (indRelRSTPO SRel TRel)
using indRelRSTPO.trans[where SRel=SRel and TRel=TRel]
unfolding trans-def
by blast
ultimately show trans (symcl (indRelRSTPO SRel TRel))
using symm-closure-of-preorder-is-trans[where Rel=indRelRSTPO SRel TRel]
by blast
qed
ultimately show \(?thesis
unfolding preorder-on-def
by blast
qed

lemma (in encoding) fully-abstract-impl-symcl-source-target-relation-is-preorder:
fixes SRel :: \("sproc \times \"sproc\") set
and TRel :: \("sproc \times \"sproc\") set
assumes fullAbs: fully-abstract ((symcl (SRel\textsuperscript=))\textsuperscript+) ((symcl (TRel\textsuperscript=))\textsuperscript+)
shows \( \exists \text{Rel}. (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}) \)
\( \wedge ((\text{symcl} (SRel\textsuperscript=))\textsuperscript+) = \{(S1, S2). (\text{SourceTerm} S1, \text{SourceTerm} S2) \in \text{symcl} \text{Rel}\} \)
\( \wedge ((\text{symcl} (TRel\textsuperscript=))\textsuperscript+) = \{(T1, T2). (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{symcl} \text{Rel}\} \)
\( \wedge \text{preorder} (\text{symcl} \text{Rel}) \)
proof –
have refl ((symcl (TRel\textsuperscript=))\textsuperscript+)
using refl-symmm-trans-closure-is-symm-refl-trans-closure[of TRel]
refl-trancl[of TRel]
unfolding sym-def refl-on-def
by auto
moreover have sym ((symcl (TRel\textsuperscript=))\textsuperscript+)
using sym-symcl[of TRel\textsuperscript=] sym-trancl[of symcl (TRel\textsuperscript=)]
by simp
moreover have trans ((symcl (TRel\textsuperscript=))\textsuperscript+)
by simp
ultimately show \(?thesis
using fully-abstract-wrt-equivalences-impl-symcl-source-target-relation-is-preorder[where SRel=(symcl (SRel\textsuperscript=))\textsuperscript+ and TRel=(symcl (TRel\textsuperscript=))\textsuperscript+] fullAbs
refl-symmm-closure-is-symm-refl-closure
unfolding preorder-on-def
by blast
qed

lemma (in encoding) fully-abstract-wrt-preorders-impl-source-target-relation-is-trans:
fixes SRel :: \("sproc \times \"sproc\") set
and TRel :: \("sproc \times \"sproc\") set
assumes fullAbs: fully-abstract SRel TRel
shows \( \exists \text{Rel}. (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}) \)
\( \wedge \text{SRel} = \{(S1, S2). (\text{SourceTerm} S1, \text{SourceTerm} S2) \in \text{Rel}\} \)
\( \wedge \text{TRel} = \{(T1, T2). (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel}\} \)
\( \wedge ((\text{refl} SRel \wedge \text{trans} TRel) \quad \leftrightarrow \quad \text{trans} (\text{Rel} \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\})) \)

proof –
define Rel where Rel = (indRelSTEQ SRel TRel) \( - \{(P, Q). \exists S. [S] \in T P \land S \in S Q\} \)
\( \cup \{(P, Q). \exists S1 S2. S1 \in S P \land S2 \in S Q \wedge (S1, S2) \notin SRel\} \)
\[ \begin{align*}
\cup \{(P, Q), \exists T1 T2. T1 \in T P \land T2 \in T Q \land (T1, T2) \notin TRel\} \\
\text{from \ Rel-def have } \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \\
\text{by (simp add: indRelSTEQ enslR \ where } SRel=SRel \text{ and } TRel=TRel) \\
\text{moreover from \ Rel-def have } SRel = \{(S1, S2). (\text{SourceTerm } S1, \text{SourceTerm } S2) \in \text{Rel}\} \\
\text{proof auto} \\
\text{fix } S1 S2 \\
\text{assume } (S1, S2) \in SRel \\
\text{thus SourceTerm } S1 \sim[\_]<SRel,TRel> SourceTerm S2 \\
\text{by (simp add: indRelSTEQ source \ where } SRel=SRel \text{ and } TRel=TRel) \\
\text{qed} \\
\text{moreover from \ Rel-def have } TRel = \{(T1, T2). (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\} \\
\text{proof auto} \\
\text{fix } T1 T2 \\
\text{assume } (T1, T2) \in TRel \\
\text{thus TargetTerm } T1 \sim[\_]<SRel,TRel> TargetTerm T2 \\
\text{by (simp add: indRelSTEQ target \ where } SRel=SRel \text{ and } TRel=TRel) \\
\text{qed} \\
\text{moreover} \\
\text{have } (\text{refl } SRel \land \text{trans } TRel) \leftrightarrow \text{trans } (\text{Rel } \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\}) \\
\text{proof (rule iffI, erule conjE) } \\
\text{assume reflS: refl } SRel \text{ and transT: trans } TRel \\
\text{have } \text{Rel } \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\} = \text{indRelSTEQ } SRel \text{ TRel} \\
\text{proof (auto simp add: Rel-def) } \\
\text{fix } S \\
\text{show TargetTerm } ([S]) \sim[\_]<SRel,TRel> SourceTerm S \\
\text{by (rule indRelSTEQ enslL) } \\
\text{next} \\
\text{fix } S1 S2 \\
\text{assume SourceTerm } S1 \sim[\_]<SRel,TRel> SourceTerm S2 \\
\text{with fullAbs reflS transT have } (S1, S2) \in SRel \\
\text{using full-abstraction-art-preorders-impl-indRelSTEQ-to-SRel-and-TRel(\text{fullAbs reflS transT}) \ where } \\
\text{SRel=SRel \ and } TRel=TRel \\
\text{by blast} \\
\text{moreover assume } (S1, S2) \notin SRel \\
\text{ultimately show False } \\
\text{by simp} \\
\text{next} \\
\text{fix } T1 T2 \\
\text{assume TargetTerm } T1 \sim[\_]<SRel,TRel> TargetTerm T2 \\
\text{with fullAbs reflS transT have } (T1, T2) \in TRel \\
\text{using full-abstraction-art-preorders-impl-indRelSTEQ-to-SRel-and-TRel(\text{fullAbs reflS transT}) \ where } \\
\text{SRel=SRel \ and } TRel=TRel \\
\text{by blast} \\
\text{moreover assume } (T1, T2) \notin TRel \\
\text{ultimately show False } \\
\text{by simp} \\
\text{qed} \\
\text{thus trans } (\text{Rel } \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\}) \\
\text{using indRelSTEQ-trans \ where } SRel=SRel \text{ and } TRel=TRel \\
\text{unfolding trans-def } \\
\text{by blast} \\
\text{next} \\
\text{assume transR: trans } (\text{Rel } \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\}) \\
\text{show refl SRel } \land \text{trans } TRel \\
\text{unfolding trans-def refl-on-def} \\
\text{proof auto} \\
\text{fix } S \\
\text{from \ Rel-def have } (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel } \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\} \\
\text{by (simp add: indRelSTEQ enslR) } \\
\text{moreover have } (\text{TargetTerm } ([S]), \text{SourceTerm } S) \in \text{Rel } \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\} \\
\text{by simp} \\
\end{align*} \]
ultimately have \((SourceTerm\ S, SourceTerm\ S)\) \(\in\) \(Rel\)
  using \(transR\)
  unfolding \(trans-def\)
  by \(blast\)
with \(Rel-def\) show \((S, S)\) \(\in\) \(SRel\)
  by \(simp\)
next
fix \(TP, TQ, TR\)
assume \((TP, TQ)\) \(\in\) \(TRel\)
with \(Rel-def\) have \((TargetTerm\ TP, TargetTerm\ TQ)\) \(\in\) \(Rel\) \(\cup\) \{(\(P, Q\), \(\exists \, S. \, [S] \in T P \land S \in S Q\))
  by \((simp\ add: indRelSTEQ.target)\)
moreover assume \((TQ, TR)\) \(\in\) \(TRel\)
with \(Rel-def\) have \((TargetTerm\ TQ, TargetTerm\ TR)\) \(\in\) \(Rel\) \(\cup\) \{(\(P, Q\), \(\exists \, S. \, [S] \in T P \land S \in S Q\))
  by \((simp\ add: indRelSTEQ.target)\)
ultimately have \((TargetTerm\ TP, TargetTerm\ TR)\) \(\in\) \(Rel\)
  using \(transR\)
  unfolding \(trans-def\)
  by \(blast\)
with \(Rel-def\) show \((TP, TR)\) \(\in\) \(TRel\)
  by \(simp\)
qed

lemma (in encoding) fully-abstract-wrt-preorders-impl-source-target-relation-is-trans-B:
fixes \(SRel\) :: \(\langle\text{proc} S \times \text{proc} S\rangle\set\)
  and \(TRel\) :: \(\langle\text{proc} T \times \text{proc} T\rangle\set\)
assumes \(fullAbs\): fully-abstract \(SRel\) \(TRel\)
  and \(reflT\): \(refl\) \(TRel\)
  and \(transT\): \(trans\) \(TRel\)
shows \(\exists \, Rel. \, (\forall \, S. \, (SourceTerm\ S, TargetTerm\ ([S])) \in Rel\)
  \(\wedge\) \(SRel\) \(=\) \{(\(S1, S2\), \(SourceTerm\ S1, SourceTerm\ S2\)) \(\in\) \(Rel\)\}
  \(\wedge\) \(TRel\) \(=\) \{(\(T1, T2\), \(TargetTerm\ T1, TargetTerm\ T2\)) \(\in\) \(Rel\)\}
  \(\wedge\) \(trans\) \(\in\) \(\{(P, Q), \, \exists \, S. \, [S] \in T P \land S \in S Q\}\)

proof –
define \(fullAbs\) where \(Rel\) \(=\) \((indRelSTEQ\ SRel\ TRel)\) \(\wedge\) \{(\(P, Q\), \(\exists \, S. \, [S] \in T P \land S \in S Q\))
from \(fullAbs\) \(reflT\) have \(reflS\) \(=\) \(refl\) \(SRel\)
  unfolding \(refl-on-def\)
  by \(auto\)
from \(Rel-def\) have \(\forall \, S. \, (SourceTerm\ S, TargetTerm\ ([S])) \in Rel\)
  by \((simp\ add: indRelSTEQ.encR[where\ SRel=SRel\ and\ TRel=TRel])\)
moreover from \(Rel-def\) have \(SRel\) \(=\) \{(\(S1, S2\), \(SourceTerm\ S1, SourceTerm\ S2\)) \(\in\) \(Rel\)\}
proof auto
fix \(S1, S2\)
assume \((S1, S2)\) \(\in\) \(SRel\)
thus \(SourceTerm\ S1 \sim \[\cdot\]_{\langle SRel, TRel\rangle} SourceTerm\ S2\)
  by \((simp\ add: indRelSTEQ.encR[where\ SRel=SRel\ and\ TRel=TRel])\)
next
fix \(S1, S2\)
assume \(SourceTerm\ S1 \sim \[\cdot\]_{\langle SRel, TRel\rangle} SourceTerm\ S2\)
with \(fullAbs\) \(transT\) \(reflS\) show \((S1, S2)\) \(\in\) \(SRel\)
  using \(full-abstract-wrt-preorders-impl-indRelSTEQ-to-SRel-and-TRel(1)[where\ SRel=SRel\ and\ TRel=TRel]\)
  by \(blast\)
qed
moreover from \(Rel-def\) have \(TRel\) \(=\) \{(\(T1, T2\), \(TargetTerm\ T1, TargetTerm\ T2\)) \(\in\) \(Rel\)\}
proof auto
fix \(T1, T2\)
assume \((T1, T2)\) \(\in\) \(TRel\)

thus \( \text{TargetTerm } T1 \sim [\_] <\text{SRel}, \text{TRel}> \text{TargetTerm } T2 \)

by (simp add: \text{indRelSTEQ,target[where SRel=SRel and TRel=TRel]})

next

fix \( T1, T2 \)

assume \( \text{TargetTerm } T1 \sim [\_] <\text{SRel}, \text{TRel}> \text{TargetTerm } T2 \)

with \text{fullAbs transT reflS show } (T1, T2) \in TRel

using \text{full-abstraction-wrt-preorders-impl-indRelSTEQ-to-SRel-and-TRel}(5)[\text{where SRel=SRel and TRel=TRel}]

by blast

qed

moreover from \text{Rel-def have } \text{Rel} \cup \{(P, Q). \exists S. \ [S] \in T P \wedge S \in S Q\} = \text{indRelSTEQ SRel TRel}

by (auto simp add: \text{indRelSTEQ,encL})

hence trans \( \text{Rel} \cup \{(P, Q). \exists S. \ [S] \in T P \wedge S \in S Q\} \)

using \text{indRelSTEQ,trans}[\text{where SRel=SRel and TRel=TRel}]

unfolding \text{trans-def}

by auto

ultimately show \text{thesis}

by blast

qed

Thus an encoding is fully abstract w.r.t. an equivalence SRel on the source and an equivalence TRel on the target iff there exists a relation that relates source terms and their literal translations, whose sym closure is a preorder such that the reduction of this sym closure to source/target terms is SRel/TRel.

\textbf{lemma (in encoding) fully-abstract-wrt-equivalences-iff-symcl-source-target-relation-is-preorder:}

\textbf{fixes SRel :: ('procS \times 'procS) set}
\textbf{and TRel :: ('procT \times 'procT) set}

\textbf{shows (fully-abstract SRel TRel \wedge equivalence TRel) =}

\textbf{(\exists Rel. (\forall S. \ (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in Rel)}
\textbf{\wedge SRel = \{(S1, S2). \ (\text{SourceTerm } S1, \text{SourceTerm } S2) \in \text{symcl Rel} \}
\textbf{\wedge TRel = \{(T1, T2). \ (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{symcl Rel} \)
\textbf{\wedge preorder (symcl Rel)})}

\textbf{proof (rule iffI)}

\textbf{assume fully-abstract SRel TRel \wedge equivalence TRel}

\textbf{thus \exists Rel. (\forall S. \ (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in Rel)}
\textbf{\wedge SRel = \{(S1, S2). \ (\text{SourceTerm } S1, \text{SourceTerm } S2) \in \text{symcl Rel} \)
\textbf{\wedge TRel = \{(T1, T2). \ (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{symcl Rel} \)
\textbf{\wedge preorder (symcl Rel))}

using \text{fully-abstract-wrt-equivalences-impl-symcl-source-target-relation-is-preorder[where SRel=SRel and TRel=TRel]}

unfolding \text{equiv-def}

by blast

next

\textbf{assume \exists Rel. (\forall S. \ (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in Rel)}
\textbf{\wedge SRel = \{(S1, S2). \ (\text{SourceTerm } S1, \text{SourceTerm } S2) \in \text{symcl Rel} \)
\textbf{\wedge TRel = \{(T1, T2). \ (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{symcl Rel} \)
\textbf{\wedge preorder (symcl Rel))}

\textbf{from this obtain Rel}

where \( \forall S. \ (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in Rel \)
\textbf{and SRel = \{(S1, S2). \ (\text{SourceTerm } S1, \text{SourceTerm } S2) \in \text{symcl Rel} \)
\textbf{and A1: TRel = \{(T1, T2). \ (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{symcl Rel} \)
\textbf{and A2: preorder (symcl Rel)}

by blast

\textbf{hence A5: fully-abstract SRel TRel}

using \text{source-target-relation-with-trans-symcl-impl-full-abstraction[where Rel=Rel]}

unfolding \text{preorder-on-def}

by blast

\textbf{moreover have equivalence TRel}

unfolding \text{trans-def equiv-def sym-def refl-on-def}

\textbf{proof auto}

fix \( T \)
from $A1 A2$ show $(T, T) \in TRel$

  unfolding preorder-on-def refl-on-def
  by blast

next

fixes $T1 T2$

assume $(T1, T2) \in TRel$

with $A1$ show $(T2, T1) \in TRel$

  unfolding trans-def preorder-on-def
  by blast

qed

ultimately show fully-abstract $SRel TRel \land$ equivalence $TRel$

by blast

qed

lemma (in encoding) fully-abstract-iff-symcl-source-target-relation-is-preorder:

  fixes $SRel :: (\text{'}procS \times \text{'}procS) \text{ set}$
  and $TRel :: (\text{'}procT \times \text{'}procT) \text{ set}$

  shows fully-abstract $((\text{symcl} \ (SRel^=))^+) \ ((\text{symcl} \ (TRel^=))^+)$ =

    $(\exists \text{rel}. \ (\forall S. \ (\text{SourceTerm S, TargetTerm } \{[S]\}) \in \text{Rel})$
    \land \ (symcl \ (SRel^=))^+ = \{(S1, S2). \ (\text{SourceTerm S1, SourceTerm S2}) \in \text{symcl Rel}\}
    \land \ (symcl \ (TRel^=))^+ = \{(T1, T2). \ (\text{TargetTerm T1, TargetTerm T2}) \in \text{symcl Rel}\}
    \land \ \text{preorder} \ (\text{symcl Rel})$

proof (rule iffI)

  assume fully-abstract $((\text{symcl} \ (SRel^=))^+) \ ((\text{symcl} \ (TRel^=))^+)$

  thus $\exists \text{rel}. \ (\forall S. \ (\text{SourceTerm S, TargetTerm } \{[S]\}) \in \text{Rel})$

    \land \ (symcl \ (SRel^=))^+ = \{(S1, S2). \ (\text{SourceTerm S1, SourceTerm S2}) \in \text{symcl Rel}\}
    \land \ (symcl \ (TRel^=))^+ = \{(T1, T2). \ (\text{TargetTerm T1, TargetTerm T2}) \in \text{symcl Rel}\}
    \land \ \text{preorder} \ (\text{symcl Rel})$

  using fully-abstract-impl-symcl-source-target-relation-is-preorder[where $SRel=SRel$ and $TRel=TRel$]

  by blast

next

assume $\exists \text{rel}. \ (\forall S. \ (\text{SourceTerm S, TargetTerm } \{[S]\}) \in \text{Rel})$

  \land \ (symcl \ (SRel^=))^+ = \{(S1, S2). \ (\text{SourceTerm S1, SourceTerm S2}) \in \text{symcl Rel}\}
  \land \ (symcl \ (TRel^=))^+ = \{(T1, T2). \ (\text{TargetTerm T1, TargetTerm T2}) \in \text{symcl Rel}\}

from this obtain Rel

where $\forall S. \ (\text{SourceTerm S, TargetTerm } \{[S]\}) \in \text{Rel}$

  and $\ (\text{symcl} \ (SRel^=))^+ = \{(S1, S2). \ (\text{SourceTerm S1, SourceTerm S2}) \in \text{symcl Rel}\}$
  and $\ (\text{symcl} \ (TRel^=))^+ = \{(T1, T2). \ (\text{TargetTerm T1, TargetTerm T2}) \in \text{symcl Rel}\}$

  and $\text{preorder} \ (\text{symcl Rel})$

by blast

thus fully-abstract $((\text{symcl} \ (SRel^=))^+) \ ((\text{symcl} \ (TRel^=))^+)$

  using source-target-relation-with-trans-symcl-impl-full-abstraction[where $Rel=Rel$]

  unfolding preorder-on-def

  by blast

qed

9.5 Full Abstraction without Relating Translations to their Source Terms

Let $Rel$ be the result of removing from indRelSTEQ all pairs of two source or two target terms that are not contained in $SRel$ or $TRel$. Then a fully abstract encoding ensures that $Rel$ is trans iff $SRel$ is refl and $TRel$ is trans.

lemma (in encoding) full-abstraction-impl-indRelSTEQ-is-trans:

  fixes $SRel :: (\text{'}procS \times \text{'}procS) \text{ set}$


227
and \( TRel := \{ 'procT \times 'procT \} \) set
and \( Rel := \{ ('procS, 'procT) Proc \times ('procS, 'procT) Proc \} \) set
assumes fullAbs: fully-abstract \( SRel \) \( TRel \)
and rel: \( Rel = ((indRelSTEQ SRel \ TRel) \)
\( \setminus \{(P, Q), (P \in ProcS \land Q \in ProcS) \lor (P \in ProcT \land Q \in ProcT)\}\)
\( \cup \{(P, Q), (\exists SP SQ, SP \in S P \land SQ \in S Q \land (SP, SQ) \in SRel) \land \forall \exists TP TQ, TP \in T P \land TQ \in T Q \land (TP, TQ) \in TRel\}\)
shows \((\text{refl} SRel \land \text{trans} TRel) = \text{trans} Rel\)
unfolding trans-def
proof auto
fix \( P, Q, R \)
assume \( A1: \text{refl} SRel \land A2: \forall x y, (x, y) \in TRel \rightarrow (\forall z, (y, z) \in TRel \rightarrow (x, z) \in TRel) \)
and \( A3: (P, Q) \in Rel \land A4: (Q, R) \in Rel \)
from fullAbs rel have \( A5: \forall SP SQ, (SourceTerm SP, SourceTerm SQ) \in Rel \rightarrow ([SP], [SQ]) \in TRel \)
by simp
from rel have \( A6: \forall TP TQ, (TargetTerm TP, TargetTerm TQ) \in Rel \rightarrow (TP, TQ) \in TRel \)
by simp
have \( A7: \forall SP TQ, (SourceTerm SP, TargetTerm TQ) \in Rel \rightarrow ([SP], TQ) \in TRel \)
proof clarify
fix \( SP TQ \)
assume \( \text{SourceTerm SP, TargetTerm TQ} \in Rel \)
with rel have \( \text{SourceTerm SP} \sim [\cdot] <SRel, TRel> \) \( \text{TargetTerm TQ} \)
by simp
with \( A1, A2 \) fullAbs show \( ([SP], TQ) \in TRel \)
using full-abstraction-wrt-pred-orders-impl-indRelSTEQ-to-SRel-and-TRel(3)[where
\( SRel=SRel \land TRel=TRel \)]
unfolding trans-def
by blast
qed
have \( A8: \forall TP SQ, (TargetTerm TP, SourceTerm SQ) \in Rel \rightarrow (TP, [SQ]) \in TRel \)
proof clarify
fix \( TP SQ \)
assume \( \text{TargetTerm TP, SourceTerm SQ} \in Rel \)
with rel have \( \text{TargetTerm TP} \sim [\cdot] <SRel, TRel> \) \( \text{SourceTerm SQ} \)
by simp
with \( A1, A2 \) fullAbs show \( (TP, [SQ]) \in TRel \)
using full-abstraction-wrt-pred-orders-impl-indRelSTEQ-to-SRel-and-TRel(4)[where
\( SRel=SRel \land TRel=TRel \)]
unfolding trans-def
by blast
qed
show \((P, R) \in Rel\)
proof (cases \( P \))
case \( \text{SourceTerm SP} \)
assume \( A9: SP \in S P \)
show \((P, R) \in Rel\)
proof (cases \( Q \))
case \( \text{SourceTerm SQ} \)
assume \( A10: SQ \in S Q \)
with \( A3, A5, A9 \) have \( A11: ([SP], [SQ]) \in TRel \)
by simp
show \((P, R) \in Rel\)
proof (cases \( R \))
case \( \text{SourceTerm SR} \)
assume \( A12: SR \in S R \)
with \( A4, A5, A10 \) have \(([SQ], [SR]) \in TRel\)
by simp
with \( A2, A11 \) have \(([SP], [SR]) \in TRel\)
by blast
with fullAbs have \((SP, SR) \in SRel\)
by simp
with rel A9 A12 show \((P, R) \in \text{Rel}\)
  by simp
next
case (TargetTerm TR)
  assume A12: \(TR \in T \ R\)
  from A9 have \(P \sim \frac{}{<SRel,TRel>} \text{TargetTerm} \left(\frac{}{\llbracket SP \rrbracket}\right)\)
    by (simp add: indRelSTEQ.encR)
  moreover from A4 A7 A10 A12 have \(\left[\llbracket SQ \rrbracket, TR\right] \in T \text{Rel}\)
    by simp
  with A2 A11 have \(\left[\llbracket SP \rrbracket, TR\right] \in T \text{Rel}\)
    by blast
  with A12 have TargetTerm \(\left[\llbracket SP \rrbracket\right] \sim \frac{}{<SRel,TRel>} R\)
    by (simp add: indRelSTEQ.target)
  ultimately have \(P \sim \frac{}{<SRel,TRel>} R\)
    by (rule indRelSTEQ.trans)
  with rel A9 A12 show \((P, R) \in \text{Rel}\)
    by simp
qed
next
case (TargetTerm TQ)
  assume A10: \(TQ \in T \ Q\)
  with A3 A7 A9 have A11: \(\left[\llbracket SP \rrbracket, TQ\right] \in T \text{Rel}\)
    by simp
  show \((P, R) \in \text{Rel}\)
    proof (cases R)
      case (SourceTerm SR)
      assume A12: \(SR \in S \ R\)
      with A4 A8 A10 have \(\left[\llbracket TQ, SR\right]\right] \in T \text{Rel}\)
        by simp
      with A2 A11 have \(\left[\llbracket SP \rrbracket, [\llbracket SR]\right]\right] \in T \text{Rel}\)
        by blast
      with fullAbs have \(SP, SR \in S \text{Rel}\)
        by simp
      with rel A9 A12 show \((P, R) \in \text{Rel}\)
        by simp
    next
      case (TargetTerm TR)
      assume A12: \(TR \in T \ R\)
      from A9 have \(P \sim \frac{}{<SRel,TRel>} \text{TargetTerm} \left(\frac{}{\llbracket SP \rrbracket}\right)\)
        by (simp add: indRelSTEQ.encR)
      moreover from A4 A6 A10 A12 have \(\left[\llbracket TQ, TR\right]\right] \in T \text{Rel}\)
        by simp
      with A2 A11 have \(\left[\llbracket SP \rrbracket, TR\right] \in T \text{Rel}\)
        by blast
      with A12 have TargetTerm \(\left[\llbracket SP \rrbracket\right] \sim \frac{}{<SRel,TRel>} R\)
        by (simp add: indRelSTEQ.target)
      ultimately have \(P \sim \frac{}{<SRel,TRel>} R\)
        by (rule indRelSTEQ.trans)
      with A9 A12 rel show \((P, R) \in \text{Rel}\)
        by simp
      qed
      qed
next
case (TargetTerm TP)
  assume A9: \(TP \in T \ P\)
  show \((P, R) \in \text{Rel}\)
    proof (cases Q)
      case (SourceTerm SQ)
      assume A10: \(SQ \in S \ Q\)
      with A3 A8 A9 have A11: \(\left[\llbracket TP, SQ\right]\right] \in T \text{Rel}\)
        by simp
    qed
show \((P, R) \in \text{Rel}\)
proof (cases \(R\))
case (SourceTerm \(SR\))
assume \(A12\): \(SR \in S R\)
with \(A4\) \(A5\) \(A10\) have \([SQ], [SR] \) \(\in \text{TRel}\)
by simp
with \(A2\) \(A11\) have \((TP, [SR]) \) \(\in \text{TRel}\)
by blast
with \(A9\) have \(P \sim [\!\!]<SRel,TRel> \text{TargetTerm} ([SR] )\)
by (simp add: \indRelSTEQ\text{.target})
moreover from \(A12\) have \(\text{TargetTerm} ([SR]) \sim [\!\!]<SRel,TRel> R\)
by (simp add: \indRelSTEQ\text{.encL})
ultimately have \(P \sim [\!\!]<SRel,TRel> R\)
by (rule \indRelSTEQ\text{.trans})
with rel \(A9\) \(A12\) show \((P, R) \in \text{Rel}\)
by simp
qed
next
case (TargetTerm \(TR\))
assume \(A12\): \(TR \in T R\)
with \(A4\) \(A7\) \(A10\) have \([SQ], TR \) \(\in \text{TRel}\)
by simp
with \(A2\) \(A11\) have \((TP, TR) \) \(\in \text{TRel}\)
by blast
with rel \(A9\) \(A12\) show \((P, R) \in \text{Rel}\)
by simp
qed
next
case (TargetTerm \(TQ\))
assume \(A10\): \(TQ \in T Q\)
with \(A3\) \(A6\) \(A9\) have \(A11\): \((TP, TQ) \) \(\in \text{TRel}\)
by simp
show \((P, R) \in \text{Rel}\)
proof (cases \(R\))
case (SourceTerm \(SR\))
assume \(A12\): \(SR \in S R\)
with \(A4\) \(A8\) \(A10\) have \((TQ, [SR]) \) \(\in \text{TRel}\)
by simp
with \(A2\) \(A11\) have \((TP, [SR]) \) \(\in \text{TRel}\)
by blast
with \(A9\) have \(P \sim [\!\!]<SRel,TRel> \text{TargetTerm} ([SR] )\)
by (simp add: \indRelSTEQ\text{.target})
moreover from \(A12\) have \(\text{TargetTerm} ([SR]) \sim [\!\!]<SRel,TRel> R\)
by (simp add: \indRelSTEQ\text{.encL})
ultimately have \(P \sim [\!\!]<SRel,TRel> R\)
by (rule \indRelSTEQ\text{.trans})
with rel \(A9\) \(A12\) show \((P, R) \in \text{Rel}\)
by simp
qed
qed
next
assume \(B\): \(\forall x y. (x, y) \in \text{Rel} \rightarrow (\forall z. (y, z) \in \text{Rel} \rightarrow (x, z) \in \text{Rel})\)
thus refl SRel
  unfolding refl-on-def
proof auto
fix S
from rel have (SourceTerm S, TargetTerm ([S])) ∈ Rel
  by (simp add: indRelSTEQ encR)
moreover from rel have (TargetTerm ([S]), SourceTerm S) ∈ Rel
  by (simp add: indRelSTEQ encL)
ultimately have (SourceTerm S, SourceTerm S) ∈ Rel
  using B
  by blast
with rel show (S, S) ∈ SRel
  by simp
qed
next
fix TP TQ TR
assume ∀ x y. (x, y) ∈ Rel ⟹ (∀ z. (y, z) ∈ Rel ⟹ (x, z) ∈ Rel)
moreover assume (TP, TQ) ∈ TRel
with rel have (TargetTerm TP, TargetTerm TQ) ∈ Rel
  by simp
moreover assume (TQ, TR) ∈ TRel
with rel have (TargetTerm TQ, TargetTerm TR) ∈ Rel
  by simp
ultimately have (TargetTerm TP, TargetTerm TR) ∈ Rel
  by blast
with rel show (TP, TR) ∈ TRel
  by simp
qed

Whenever an encoding induces a trans relation that includes SRel and TRel and relates source terms
to their literal translations in both directions, the encoding is fully abstract w.r.t. SRel and TRel.

lemma (in encoding) trans-source-target-relation-impl-fully-abstract:
  fixes Rel :: ((procS, ’procT) Proc × (procS, procT) Proc) set
  and SRel :: (procS × procS) set
  and TRel :: (procT × procT) set
  assumes enc: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
    ∧ (TargetTerm ([S]), SourceTerm S) ∈ Rel
  and srel: SRel = {(S1, S2). (SourceTerm S1, SourceTerm S2) ∈ Rel}
  and trel: TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
  and trans: trans Rel
  shows fully-abstract SRel TRel
proof auto
fix S1 S2
assume (S1, S2) ∈ SRel
with srel have (SourceTerm S1, SourceTerm S2) ∈ Rel
  by simp
with enc trans have (TargetTerm ([S1]), TargetTerm ([S2])) ∈ Rel
  unfolding trans-def
  by blast
with trel show ([S1], [S2]) ∈ TRel
  by simp
next
fix S1 S2
assume ([S1], [S2]) ∈ TRel
with trel have (TargetTerm ([S1]), TargetTerm ([S2])) ∈ Rel
  by simp
with enc trans have (SourceTerm S1, SourceTerm S2) ∈ Rel
  unfolding trans-def
  by blast
with srel show (S1, S2) ∈ SRel
Assume TRel is a preorder. Then an encoding is fully abstract w.r.t. SRel and TRel iff there exists a relation that relates add least all source terms to their literal translations, includes SRel and TRel, and whose union with the relation that relates exactly all literal translations to their source terms is trans.

lemma (in encoding) source-target-relation-with-trans-impl-full-abstraction:
  fixes Rel :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) set
  assumes enc: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
          and trans: trans (Rel ∪ {(P, Q), ∃ S. [S] ∈ T P ∧ S ∈ S Q})
  shows fully-abstract ((S1, S2), (SourceTerm S1, SourceTerm S2) ∈ Rel)
                    ((T1, T2), (TargetTerm T1, TargetTerm T2) ∈ Rel)

proof auto
  fix S1 S2
  define Rel' where Rel' = Rel ∪ {(P, Q), ∃ S. [S] ∈ T P ∧ S ∈ S Q}
  from Rel'-def have (TargetTerm ([S1]), SourceTerm S1) ∈ Rel'
    by simp
  moreover assume (SourceTerm S1, SourceTerm S2) ∈ Rel
  with Rel'-def have (SourceTerm S1, SourceTerm S2) ∈ Rel'
    by simp
  moreover from enc Rel'-def have (SourceTerm S2, TargetTerm ([S2])) ∈ Rel'
    by simp
  ultimately show (TargetTerm ([S1]), TargetTerm ([S2])) ∈ Rel
    using trans Rel'-def
    unfolding trans-def
    by blast

next
  fix S1 S2
  define Rel' where Rel' = Rel ∪ {(P, Q), ∃ S. [S] ∈ T P ∧ S ∈ S Q}
  from enc Rel'-def have (SourceTerm S1, TargetTerm ([S1])) ∈ Rel'
    by simp
  moreover assume (TargetTerm ([S1]), TargetTerm ([S2])) ∈ Rel
  with Rel'-def have (TargetTerm ([S1]), TargetTerm ([S2])) ∈ Rel'
    by simp
  moreover from Rel'-def have (TargetTerm ([S2]), SourceTerm S2) ∈ Rel'
    by simp
  ultimately show (SourceTerm S1, SourceTerm S2) ∈ Rel
    using trans Rel'-def
    unfolding trans-def
    by blast

qed

lemma (in encoding) fully-abstract-wrt-preorders-iff-source-target-relation-is-transB:
  fixes SRel :: ('procS × 'procS) set
  and TRel :: ('procT × 'procT) set
  assumes preord: preorder TRel
  shows fully-abstract SRel TRel =
           (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
                ∧ SRel = {(S1, S2), (SourceTerm S1, SourceTerm S2) ∈ Rel}
                ∧ TRel = {(T1, T2), (TargetTerm T1, TargetTerm T2) ∈ Rel}
                ∧ trans (Rel ∪ {(P, Q), ∃ S. [S] ∈ T P ∧ S ∈ S Q}))

proof (rule iffI)
  assume fully-abstract SRel TRel
  with preord show ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
                             ∧ SRel = {(S1, S2), (SourceTerm S1, SourceTerm S2) ∈ Rel}
                             ∧ TRel = {(T1, T2), (TargetTerm T1, TargetTerm T2) ∈ Rel}
                             ∧ trans (Rel ∪ {(P, Q), ∃ S. [S] ∈ T P ∧ S ∈ S Q})
    using fully-abstract-wrt-preorders-impl-source-target-relation-is-trans[where SRel=SRel
                                                                 and TRel=TRel]
unfolding preorder-on-def refl-on-def

by auto

next

assume \( \exists \mathsf{Rel}. (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \mathsf{Rel}) \)
\( \land \mathsf{SRel} = \{(S1, S2). (\text{SourceTerm} S1, \text{SourceTerm} S2) \in \mathsf{Rel}\} \)
\( \land \mathsf{TRel} = \{(T1, T2). (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \mathsf{Rel}\} \)
\( \land \text{trans} ([\mathsf{Rel} \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\}]) \)

from this obtain \( \mathsf{Rel} \)

where \( \forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \mathsf{Rel} \)

and \( \mathsf{SRel} = \{(S1, S2). (\text{SourceTerm} S1, \text{SourceTerm} S2) \in \mathsf{Rel}\} \)

and \( \mathsf{TRel} = \{(T1, T2). (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \mathsf{Rel}\} \)

and \( \text{trans} ([\mathsf{Rel} \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\}]) \)

by blast

thus fully-abstract SRel TRel

using source-target-relation-with-trans-impl-full-abstraction[where \mathsf{Rel=}\mathsf{Rel}] by blast

qed

The same holds if to obtain transitivity the union may contain additional pairs that do neither relate two source nor two target terms.

lemma (in encoding) fully-abstract-wrt-preorders-iff-source-target-relation-union-is-trans:

fixes \( \mathsf{SRel} ::= (\text{procS} \times \text{procS}) \) set

and \( \mathsf{TRel} ::= (\text{procT} \times \text{procT}) \) set

shows (fully-abstract SRel TRel \( \land \) refl SRel \( \land \) trans TRel) =

(\( \exists \mathsf{Rel}. (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \mathsf{Rel}) \)
\( \land \mathsf{SRel} = \{(S1, S2). (\text{SourceTerm} S1, \text{SourceTerm} S2) \in \mathsf{Rel}\} \)
\( \land \mathsf{TRel} = \{(T1, T2). (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \mathsf{Rel}\} \)
\( \land (\exists \mathsf{Rel}'. (\forall (P, Q) \in \mathsf{Rel}'. P \in \text{ProcS} \longleftrightarrow Q \in \text{ProcT}) \)
\( \land \text{trans} ([\mathsf{Rel} \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\} \cup \mathsf{Rel}'\}) \)

proof (rule iffI, (erule conjI)+)

assume fully-abstract SRel TRel and refl SRel and trans TRel

from this obtain \( \mathsf{Rel} \) where \( A1: \forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \mathsf{Rel} \)

and \( A2: \mathsf{SRel} = \{(S1, S2). (\text{SourceTerm} S1, \text{SourceTerm} S2) \in \mathsf{Rel}\} \)

and \( A3: \mathsf{TRel} = \{(T1, T2). (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \mathsf{Rel}\} \)

and \( A4: \text{trans} ([\mathsf{Rel} \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\} \cup \mathsf{Rel}'\}) \)

using fully-abstract-wrt-preorders-impl-source-target-relation-is-trans[where \mathsf{SRel=}\mathsf{SRel} and \mathsf{TRel=}\mathsf{TRel}]

by blast

have \( \forall (P, Q) \in \{\}. P \in \text{ProcS} \longleftrightarrow Q \in \text{ProcT} \)

by simp

moreover from \( A4 \) have \( \text{trans} ([\mathsf{Rel} \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\} \cup \{\}) \)

unfolding trans-def

by blast

ultimately show \( \exists \mathsf{Rel}. (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \mathsf{Rel}) \)
\( \land \mathsf{SRel} = \{(S1, S2). (\text{SourceTerm} S1, \text{SourceTerm} S2) \in \mathsf{Rel}\} \)
\( \land \mathsf{TRel} = \{(T1, T2). (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \mathsf{Rel}\} \)
\( \land (\exists \mathsf{Rel}'. (\forall (P, Q) \in \mathsf{Rel}'. P \in \text{ProcS} \longleftrightarrow Q \in \text{ProcT}) \)
\( \land \text{trans} ([\mathsf{Rel} \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\} \cup \mathsf{Rel}'\}) \)

using \( A1 \) \( A2 \) \( A3 \)

by blast

next

assume \( \exists \mathsf{Rel}. (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \mathsf{Rel}) \)
\( \land \mathsf{SRel} = \{(S1, S2). (\text{SourceTerm} S1, \text{SourceTerm} S2) \in \mathsf{Rel}\} \)
\( \land \mathsf{TRel} = \{(T1, T2). (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \mathsf{Rel}\} \)
\( \land (\exists \mathsf{Rel}'. (\forall (P, Q) \in \mathsf{Rel}'. P \in \text{ProcS} \longleftrightarrow Q \in \text{ProcT}) \)
\( \land \text{trans} ([\mathsf{Rel} \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\} \cup \mathsf{Rel}'\}) \)

from this obtain \( \mathsf{Rel}' \)

where \( B1: \forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \mathsf{Rel} \)

and \( B2: \mathsf{SRel} = \{(S1, S2). (\text{SourceTerm} S1, \text{SourceTerm} S2) \in \mathsf{Rel}\} \)

and \( B3: \mathsf{TRel} = \{(T1, T2). (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \mathsf{Rel}\} \)
and B4: $\forall (P, Q) \in \text{Rel}'$, $P \in \text{ProcS} \leftrightarrow Q \in \text{ProcT}$
and B5: $\text{trans}(\text{Rel} \cup \{(P, Q). \exists S. \ [S] \in T \land S \in S \land Q \cup \text{Rel}'\})$

by blast
have fully-abstract SRel TRel
proof auto
fix S1 S2
have $(\text{TargetTerm} ([S1]), \text{SourceTerm} S1) \in \text{Rel} \cup \{(P, Q). \exists S. \ [S] \in T \land S \in S \land Q \cup \text{Rel}'\}$
  by simp
moreover assume $(S1, S2) \in \text{SRel}$
with B2 have $(\text{SourceTerm} S1, \text{SourceTerm} S2) \in \text{Rel} \cup \{(P, Q). \exists S. \ [S] \in T \land S \in S \land Q \cup \text{Rel}'\}$
  by simp
moreover from B1
have $(\text{SourceTerm} S2, \text{TargetTerm} ([S2])) \in \text{Rel} \cup \{(P, Q). \exists S. \ [S] \in T \land S \in S \land Q \cup \text{Rel}'\}$
  by simp
ultimately have $(\text{TargetTerm} ([S1]), \text{TargetTerm} ([S2])) \in \text{Rel} \cup \text{Rel}'$
    using B5
    unfolding trans-def
  by blast
with B3 B4 show $(S1, S2) \in \text{TRel}$
  by blast
next
fix S1 S2
from B1
have $(\text{SourceTerm} S1, \text{TargetTerm} ([S1])) \in \text{Rel} \cup \{(P, Q). \exists S. \ [S] \in T \land S \in S \land Q \cup \text{Rel}'\}$
  by simp
moreover assume $(S1, S2) \in \text{TRel}$
with B3
have $(\text{TargetTerm} ([S1]), \text{TargetTerm} ([S2])) \in \text{Rel} \cup \{(P, Q). \exists S. \ [S] \in T \land S \in S \land Q \cup \text{Rel}'\}$
  by simp
ultimately have $(\text{SourceTerm} S1, \text{SourceTerm} S2) \in \text{Rel} \cup \text{Rel}'$
    using B5
    unfolding trans-def
  by blast
with B2 B4 show $(S1, S2) \in \text{SRel}$
  by blast
qed
moreover have refl SRel
unfolding refl-on-def
proof auto
fix S
from B1 have $(\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel} \cup \{(P, Q). \exists S. \ [S] \in T \land S \in S \land Q \cup \text{Rel}'\}$
  by simp
moreover
have $(\text{TargetTerm} ([S]), \text{SourceTerm} S) \in \text{Rel} \cup \{(P, Q). \exists S. \ [S] \in T \land S \in S \land Q \cup \text{Rel}'\}$
  by simp
ultimately have $(\text{SourceTerm} S, \text{SourceTerm} S) \in \text{Rel} \cup \text{Rel}'$
    using B5
    unfolding trans-def
  by blast
with B2 B4 show $(S, S) \in \text{SRel}$
  by blast
qed
moreover have trans TRel
unfolding trans-def
proof clarify
fix TP TQ TR
assume $(TP, TQ) \in \text{TRel}$ and $(TQ, TR) \in \text{TRel}$
with B3 B4 B5 show $(TP, TR) \in \text{TRel}$
10 Combining Criteria

So far we considered the effect of single criteria on encodings. Often the quality of an encoding is prescribed by a set of different criteria. In the following we analyse the combined effect of criteria. This way we can compare criteria as well as identify side effects that result from combinations of criteria. We start with some technical lemmata. To combine the effect of different criteria we combine the conditions they induce. If their effect can be described by a predicate on the pairs of the relation, as in the case of success sensitiveness or divergence reflection, combining the effects is simple.

lemma (in encoding) criterion-iff-source-target-relation-impl-indRelR:
  fixes Cond :: ('procS ⇒ 'procT) ⇒ bool
  assumes Cond enc = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S]])) ∈ Rel) ∧ Pred Rel
  shows Cond enc = (∃ Rel'. Pred (indRelR ∪ Rel'))
proof (rule iffI)
  assume Cond enc
  with asms obtain Rel where A1: ∀ S. (SourceTerm S, TargetTerm ([S]])) ∈ Rel and A2: Pred Rel
  by blast
  from A1 have Rel = indRelR ∪ (Rel − indRelR)
  by (auto simp add: indRelR.simps)
  with A2 have Pred (indRelR ∪ (Rel − indRelR))
  by simp
  thus ∃ Rel'. Pred (indRelR ∪ Rel')
  by blast
next
  assume ∃ Rel'. Pred (indRelR ∪ Rel')
  from this obtain Rel' where Pred (indRelR ∪ Rel')
  by blast
  moreover have ∀ S. (SourceTerm S, TargetTerm ([S]])) ∈ (indRelR ∪ Rel')
  by (simp add: indRelR.encR)
  ultimately show Cond enc
  using asms
  by blast
qed

lemma (in encoding) combine-conditions-on-pairs-of-relations:
  fixes RelA RelB :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) set
  assumes ∀ (P, Q) ∈ RelA. CondA (P, Q)
  and ∀ (P, Q) ∈ RelB. CondB (P, Q)
  shows (∃ (P, Q) ∈ RelA ∩ RelB. CondA (P, Q)) ∧ (∃ (P, Q) ∈ RelA ∩ RelB. CondB (P, Q))
  using asms
  by blast

lemma (in encoding) combine-conditions-on-sets-of-relations:
  fixes RelA :: (('procS, 'procT) Proc × ('procS, 'procT) Proc) set
  unfolds trans-def
  by blast
  qed
  ultimately show fully-abstract SRel TRel ∧ refl SRel ∧ trans TRel
  by blast
  qed
lemma (in encoding) combine-conditions-on-sets-and-pairs-of-relations:

fixes RelA RelB :: (('procS', 'procT) Proc × ('procS', 'procT) Proc) set
and Cond :: (('procS', 'procT) Proc × ('procS', 'procT) Proc) set ⇒ bool
and CondA CondB :: (('procS', 'procT) Proc × ('procS', 'procT) Proc) ⇒ bool

assumes ∀ (P, Q) ∈ RelA: CondA (P, Q)
and A2: ∀ (P, Q) ∈ RelA: CondA (P, Q)
and A3: ∀ (P, Q) ∈ RelB: CondB (P, Q)
and A4: ∀ (P, Q) ∈ RelB: CondB (P, Q)

shows Cond Rel ∧ Rel ⊆ RelA ∧ Rel ⊆ RelB

using assms
by blast

lemma (in encoding) combine-conditions-on-relations-indRelR:

fixes RelA RelB :: (('procS', 'procT) Proc × ('procS', 'procT) Proc) set
and Cond :: (('procS', 'procT) Proc × ('procS', 'procT) Proc) set ⇒ bool
and CondA CondB :: (('procS', 'procT) Proc × ('procS', 'procT) Proc) ⇒ bool

assumes A1: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ RelA
and A2: ∀ (P, Q) ∈ RelA: CondA (P, Q)
and A3: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ RelB
and A4: ∀ (P, Q) ∈ RelB: CondB (P, Q)

shows Cond indRelR ⇒ (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel) ∧ (∀ (P, Q) ∈ Rel. CondA (P, Q)) ∧ (∀ (P, Q) ∈ Rel. CondB (P, Q)) ∧ Cond Rel)

proof

have A5: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ indRelR
by (simp add: indRelR.encR)
moreover have A6: indRelR ⊆ RelA

fix P Q
assume (P, Q) ∈ indRelR
from this A1 show (P, Q) ∈ RelA
by (induct, simp)

qed

moreover have A7: indRelR ⊆ RelB

fix P Q
assume (P, Q) ∈ indRelR
from this A3 show (P, Q) ∈ RelB
by (induct, simp)

qed

ultimately show 3 Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
∧ (∀ (P, Q) ∈ Rel. CondA (P, Q)) ∧ (∀ (P, Q) ∈ Rel. CondB (P, Q))

using combine-conditions-on-sets-and-pairs-of-relations[where RelA=RelA and RelB=RelB]
and CondA=CondA and CondB=CondB and Rel=indRelR
and Cond=λ R. ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ R] A2 A4

by blast
from A2 A4 A5 A6 A7
show Cond indRelR ⇒ (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
∧ (∀ (P, Q) ∈ Rel. CondA (P, Q)) ∧ (∀ (P, Q) ∈ Rel. CondB (P, Q)) ∧ Cond Rel)

using combine-conditions-on-sets-and-pairs-of-relations[where RelA=RelA and RelB=RelB]
and CondA=CondA and CondB=CondB and Rel=indRelR

236
and Cond=$\lambda R. \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in R \land \text{Cond } R$

by blast

qed

lemma (in encoding) indRelR-cond-respects-predA-and-reflects-predB:

fixes PredA PredB :: ('procS, 'procT) Proc \Rightarrow bool

shows \(\exists \text{Rel}. \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \land \text{rel-respects-pred Rel PredA}\)
\(\land (\exists \text{Rel}. \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \land \text{rel-reflects-pred Rel PredB})\)
= \(\exists \text{Rel}. \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \land \text{rel-respects-pred Rel PredA}\)
\(\land \text{rel-reflects-pred Rel PredB}\)

proof (rule iffI, erule conjE)

assume \(\exists \text{Rel}. \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \land \text{rel-respects-pred Rel PredA}\)

from this obtain RelA where \(A1: \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{RelA}\)
\(\land A2: \text{rel-respects-pred RelA PredA}\)

by blast

assume \(\exists \text{Rel}. \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \land \text{rel-reflects-pred Rel PredB}\)

from this obtain RelB where \(A3: \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{RelB}\)
\(\land A4: \text{rel-reflects-pred RelB PredB}\)

by blast

from A2 have \(\forall (P, Q) \in \text{RelA}. \text{PredA } P \leftrightarrow \text{PredA } Q\)

by blast

moreover from A4 have \(\forall (P, Q) \in \text{RelB}. \text{PredB } Q \rightarrow \text{PredB } P\)

by blast

ultimately have \(\exists \text{Rel}. \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}\)
\(\land (\forall (P, Q) \in \text{RelA}. \text{PredA } P = \text{PredA } Q) \land (\forall (P, Q) \in \text{RelB}. \text{PredB } Q \rightarrow \text{PredB } P)\)

using combine-conditions-on-relations-indRelR(1)\(\text{where RelA=RelA }\land \text{ RelB=RelB }\land \text{ CondA} = \lambda(P, Q). \text{PredA } P \leftrightarrow \text{PredA } Q\) and \text{CondB} = \lambda(P, Q). \text{PredB } Q \rightarrow \text{PredB } P\] A1 A3

by simp

thus \(\exists \text{Rel}. \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \land \text{rel-respects-pred Rel PredA}\)
\(\land \text{rel-reflects-pred Rel PredB}\)

by blast

next

assume \(\exists \text{Rel}. \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \land \text{rel-respects-pred Rel PredA}\)
\(\land \text{rel-reflects-pred Rel PredB}\)

thus \(\exists \text{Rel}. \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \land \text{rel-reflects-pred Rel PredA}\)
\(\land (\exists \text{Rel}. \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \land \text{rel-reflects-pred Rel PredB})\)

by blast

qed

10.1 Divergence Reflection and Success Sensitiveness

We combine results on divergence reflection and success sensitiveness to analyse their combined effect on an encoding function. An encoding is success sensitive and reflects divergence iff there exists a relation that relates source terms and their literal translations that reflects divergence and respects success.

lemma (in encoding-wrt-barbs) WSS-DR-iff-source-target-rel:

fixes success :: 'barbs

shows \((\text{enc-weakly-respects-barb-set } \{\text{success}\} \land \text{enc-reflects-divergence})\)
= \((\exists \text{Rel}. \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}\)
\(\land \text{rel-weakly-respects-barb-set Rel } (\text{STCalWB } \text{SWB } \text{TWB}) \{\text{success}\}\)
\(\land \text{rel-reflects-divergence Rel } (\text{STCal Source Target})\)

proof

have \(\forall \text{Rel}. \text{rel-reflects-divergence Rel } (\text{STCal Source Target})\)
= \(\text{rel-reflects-pred Rel } \text{divergentST}\)

by (simp add: divergentST-STCal-divergent)

moreover have \(\forall \text{Rel}. (\text{rel-weakly-respects-barb-set Rel } (\text{STCalWB } \text{SWB } \text{TWB}) \{\text{success}\})\)
= \(\text{rel-respects-pred Rel } (\lambda P. P\{\text{success}\})\)

by (simp add: STCalWB-reachesBarbST)

ultimately show \((\text{enc-weakly-respects-barb-set } \{\text{success}\} \land \text{enc-reflects-divergence})\)
= \((\exists \text{Rel}. \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel})\)
\[ \forall \rel \text{ respects - divergent } \rel \text{ (STCal Source Target)} \]
\[ \text{using success-sensitive-iff-source-target-rel-weakly-respects-success} \]
\[ \text{indRelR-cond-respects-predA-and-reflects-predB} \]
\[ \text{where} \]
\[ \PredA = \lambda P. P \downarrow \text{success and} \ PredB = \text{divergentST} \]
\[ \text{by simp} \]
\[ \text{qed} \]

**Lemma (in encoding-wrt-barbs)** \text{SS-DR-iff-source-target-rel}:

- **fixes** \text{success :: 'bars} 
- **shows** \text{enc-reflects-barb-set} \{\text{success}\} \wedge \text{enc-reflects-divergence}

\[ = (\exists \rel. (\forall S. \text{(SourceTerm } S, \text{TargetTerm } ([S])) \in \rel) \]
\[ \wedge \text{rel-respects-barb-set-rel} \text{ (STCalWB SWB TWB)} \{\text{success}\} \]
\[ \wedge \text{rel-reflects-divergence-rel} \text{ (STCal Source Target)} \]

**proof**
- **have** \( \forall \rel. \text{rel-reflects-divergence-rel} \text{ (STCal Source Target)} \)
  \[ = \text{rel-reflects-pred-rel} \text{ divergentST} \]
  
  **by** \text{(simp add: divergentST-STCal-divergent)}

- **moreover have** \( \forall \rel. \text{(rel-respects-barb-set-rel} \text{ (STCalWB SWB TWB)} \{\text{success}\} \)
  \[ = \text{rel-reflects-pred-rel} \text{ (}\lambda P. P \downarrow \text{success)} \]
  
  **by** \text{(simp add: STCalWB-hasBarbST)}

- **ultimately show** \text{enc-reflects-barb-set} \{\text{success}\} \wedge \text{enc-reflects-divergence}

\[ = (\exists \rel. (\forall S. \text{(SourceTerm } S, \text{TargetTerm } ([S])) \in \rel) \]
\[ \wedge \text{rel-respects-barb-set-rel} \text{ (STCalWB SWB TWB)} \{\text{success}\} \]
\[ \wedge \text{rel-reflects-divergence-rel} \text{ (STCal Source Target)} \]

**using success-sensitive-iff-source-target-rel-weakly-respects-success} \]
\[ \text{divergence-reflection-iff-source-target-rel-reflects-divergence} \]
\[ \text{indRelR-cond-respects-predA-and-reflects-predB} \]
\[ \text{where} \]
\[ \PredA = \lambda P. P \downarrow \text{success and} \ PredB = \text{divergentST} \]
\[ \text{by simp} \]
\[ \text{qed} \]

### 10.2 Adding Operational Correspondence

The effect of operational correspondence includes conditions (TRel is included, transitivity) that require a witness like indRelRTPO. In order to combine operational correspondence with success sensitiveness, we show that if the encoding and TRel (weakly) respects bars than indRelRTPO (weakly) respects bars. Since success is only a specific kind of bars, the same holds for success sensitiveness.

**Lemma (in encoding-wrt-barbs)** \text{enc-and-TRel-impl-indRelRTPO-weakly-respects-success}:

- **fixes** \text{success :: 'bars} 
- **and** \text{TRel :: ('procT × 'procT) set} 
- **assumes** \text{encRS: enc-weakly-respects-barb-set} \{\text{success}\} 
- **and** \text{trePS: rel-weakly-preserves-barb-set TRel TWB} \{\text{success}\} 
- **and** \text{treRS: rel-weakly-reflects-barb-set TRel TWB} \{\text{success}\} 

**shows** \text{rel-weakly-respects-barb-set (indRelRTPO TRel) (STCalWB SWB TWB) \{success\}}

**proof**
- **fix** \( P \overset{RT}{\leadsto} Q \) \text{ and} \( P \rightsquigarrow \text{(Calculus (STCalWB SWB TWB))} P' \)
- **assume** \( P \overset{\text{success}}{\leadsto} \text{STCalWB SWB TWB} \)
- **thus** \( Q \overset{\text{success}}{\leadsto} \text{STCalWB SWB TWB} \)

**proof** \text{(induct arbitrary: P')} 
- **case** \text{(encR S)} 
- **assume** \text{SourceTerm S \rightsquigarrow \text{(Calculus (STCalWB SWB TWB))} P' and P' \overset{\text{success}}{\leadsto} \text{STCalWB SWB TWB} \)
- **hence** \( S \overset{\text{success}}{\leftarrow} \text{STCalWB-reachesBarbST} \)
  
  **by** \text{blast}
- **with** \text{encRS have} \( [S] \overset{\text{success}}{\leadsto} \text{STCalWB SWB TWB} \)
  
  **by** \text{simp}
- **thus** \text{TargetTerm ([S]) \overset{\text{success}}{\leftarrow} \text{STCalWB SWB TWB} \)

238
using \texttt{STCalWB-reachesBarbST}
by \texttt{blast}

next
\begin{description}
\item[case (source S)]
assume \texttt{SourceTerm S \rightarrow (Calculus (STCalWB SWB TWB))\* P' and P'\downarrow<STCalWB SWB TWB>success}
thus \texttt{SourceTerm S\downarrow<STCalWB SWB TWB>success}
by \texttt{blast}
\end{description}

next
\begin{description}
\item[case (target T1 T2)]
assume \((T1, T2) \in TRel\)
moreover assume \texttt{TargetTerm T1 \rightarrow (Calculus (STCalWB SWB TWB))\* P'}
and \texttt{P'\downarrow<STCalWB SWB TWB>success}
hence \texttt{T1\downarrow<TWB>success}
using \texttt{STCalWB-reachesBarbST}
by \texttt{blast}
ultimately have \texttt{T2\downarrow<TWB>success}
using \texttt{trelPS}
by \texttt{simp}
thus \texttt{T1\downarrow<STCalWB SWB TWB>success}
\end{description}

next
\begin{description}
\item[case (trans P Q R)]
assume \texttt{P \rightarrow (Calculus (STCalWB SWB TWB))\* P' and P'\downarrow<STCalWB SWB TWB>success}
and \texttt{\bigwedge P', P \rightarrow (Calculus (STCalWB SWB TWB))\* P' \implies P'\downarrow<STCalWB SWB TWB>success}
hence \texttt{Q\downarrow<STCalWB SWB TWB>success}
by \texttt{simp}
moreover assume \texttt{\bigwedge Q'. Q \rightarrow (Calculus (STCalWB SWB TWB))\* Q' \implies Q'\downarrow<STCalWB SWB TWB>success}
thus \texttt{R\downarrow<STCalWB SWB TWB>success}
proof (induct arbitrary: Q')
\item[case (encR S)]
assume \texttt{TargetTerm ([S]) \rightarrow (Calculus (STCalWB SWB TWB))\* Q'}
and \texttt{Q'\downarrow<STCalWB SWB TWB>success}
hence \texttt{[S]\downarrow<TWB>success}
using \texttt{STCalWB-reachesBarbST}
by \texttt{blast}
with \texttt{encRS} have \texttt{S\downarrow<SWB>success}
by \texttt{simp}
thus \texttt{SourceTerm S\downarrow<STCalWB SWB TWB>success}
using \texttt{STCalWB-reachesBarbST}
by \texttt{blast}
\item[case (source S)]
assume \texttt{SourceTerm S \rightarrow (Calculus (STCalWB SWB TWB))\* Q' and Q'\downarrow<STCalWB SWB TWB>success}
thus \texttt{SourceTerm S\downarrow<STCalWB SWB TWB>success}
by \texttt{blast}
\item[case (target T1 T2)]
assume \((T1, T2) \in TRel\)
moreover assume \texttt{TargetTerm T2 \rightarrow (Calculus (STCalWB SWB TWB))\* Q'}
and \texttt{Q'\downarrow<STCalWB SWB TWB>success}
\end{description}
hence $T2\Downarrow <TWB>success$
   using $STCalWB$-reachesBarbST
   by blast

ultimately have $T1\Downarrow <TWB>success$
   using $trelRS$
   by blast

thus TargetTerm $T1\Downarrow <STCalWB SWB TWB>success$
   using $STCalWB$-reachesBarbST
   by blast

next
case (trans $P Q R R'$)
assume $R \mapsto \rightarrow (Calculus (STCalWB SWB TWB)) * R'$ and $R'\Downarrow <STCalWB SWB TWB>success$
and $\wedge R'. R \mapsto \rightarrow (Calculus (STCalWB SWB TWB)) * R' \Rightarrow R''\Downarrow <STCalWB SWB TWB>success$

hence $Q\Downarrow <STCalWB SWB TWB>success$
   by simp
   moreover assume $\wedge Q'. Q \mapsto \rightarrow (Calculus (STCalWB SWB TWB)) * Q' \Rightarrow Q''\Downarrow <STCalWB SWB TWB>success$
   by blast

ultimately show $P\Downarrow <STCalWB SWB TWB>success$
   by blast

qed

lemma (in encoding-wrt-barbs) enc-and-TRel-impl-indRelRTPO-weakly-respects-barbs:
fixes $TRel:: (\procT \times \procT)$ set
assumes $encRS:: enc$-weakly-respects-barbs
   and $trelPS:: rel$-weakly-preserves-barbs $TRel$ $TWB$
   and $trelRS:: rel$-weakly-reflects-barbs $TRel$ $TWB$
shows $rel$-weakly-respects-barbs ($indRelRTPO$ $TRel$) ($STCalWB SWB TWB$)
proof auto
fix $P Q x P'$
assume $P \leq \llbracket RT \rrbracket <TRel> Q$ and $P \mapsto \rightarrow (Calculus (STCalWB SWB TWB)) * P'$
and $P'\Downarrow <STCalWB SWB TWB> x$
thus $Q\Downarrow <STCalWB SWB TWB> x$

proof (induct arbitrary: $P'$)
case (encR $S$)
assume SourceTerm $S \mapsto \rightarrow (Calculus (STCalWB SWB TWB)) * P'$ and $P'\Downarrow <STCalWB SWB TWB> x$

hence $S\Downarrow <SWB> x$
   using $STCalWB$-reachesBarbST
   by blast
with $encRS$ have $\llbracket S \rrbracket \Downarrow <TWB> x$
   by simp
thus TargetTerm $\llbracket [S] \rrbracket \Downarrow <STCalWB SWB TWB> x$
   using $STCalWB$-reachesBarbST
   by blast

next
case (source $S$)
assume SourceTerm $S \mapsto \rightarrow (Calculus (STCalWB SWB TWB)) * P'$ and $P'\Downarrow <STCalWB SWB TWB> x$
thus SourceTerm $S\Downarrow <STCalWB SWB TWB> x$
   by blast

next
case (target $T1 T2$)
assume $(T1, T2) \in TRel$
moreover assume TargetTerm $T1 \mapsto \rightarrow (Calculus (STCalWB SWB TWB)) * P'$
   and $P'\Downarrow <STCalWB SWB TWB> x$

hence $T1\Downarrow <TWB> x$
   using $STCalWB$-reachesBarbST
   by blast

ultimately have $T2\Downarrow <TWB> x$
   using $trelPS$
by simp
thus TargetTerm T2\Downarrow<STCalWB SWB TWB>x
  using STCalWB-reachesBarbST
by blast

next
case (trans P Q R)
  assume P \to\to(\text{Calculus}(STCalWB SWB TWB)) \ast P' and P'\Downarrow<STCalWB SWB TWB>x
  and \land P'. P \to\to(\text{Calculus}(STCalWB SWB TWB)) \to P' \to P'\Downarrow<STCalWB SWB TWB>x
  \implies Q\Downarrow<STCalWB SWB TWB>x
  hence Q\Downarrow<STCalWB SWB TWB>x
  by simp
moreover assume \land Q'. Q \to\to(\text{Calculus}(STCalWB SWB TWB)) \to Q' \implies Q'\Downarrow<STCalWB SWB TWB>x
  \implies R\Downarrow<STCalWB SWB TWB>x
ultimately show R\Downarrow<STCalWB SWB TWB>x
by blast
qed

next
fix P Q x Q'
assume P \Downarrow<\text{RT}<\text{TRel}> Q and Q \to\to(\text{Calculus}(STCalWB SWB TWB)) \ast Q'
  and Q'\Downarrow<STCalWB SWB TWB>x
thus P\Downarrow<STCalWB SWB TWB>x

proof (induct arbitrary: Q')
case (encR S)
  assume TargetTerm ([S]) \to\to(\text{Calculus}(STCalWB SWB TWB)) \ast Q'
    and Q'\Downarrow<STCalWB SWB TWB>x
  hence [S]\Downarrow<TWB>x
    using STCalWB-reachesBarbST
    by blast
  with encRS have S\Downarrow<SWB>x
    by simp
  thus SourceTerm S\Downarrow<STCalWB SWB TWB>x
    using STCalWB-reachesBarbST
    by blast

next
case (source S)
  assume SourceTerm S \to\to(\text{Calculus}(STCalWB SWB TWB)) \ast Q' and Q'\Downarrow<STCalWB SWB TWB>x
  thus SourceTerm S\Downarrow<STCalWB SWB TWB>x
  by blast

next
case (target T1 T2)
  assume (T1, T2) \in \text{TRel}
moreover assume TargetTerm T2 \to\to(\text{Calculus}(STCalWB SWB TWB)) \ast Q'
    and Q'\Downarrow<STCalWB SWB TWB>x
  hence T2\Downarrow<TWB>x
    using STCalWB-reachesBarbST
    by blast
ultimately have T1\Downarrow<TWB>x
    using trelRS
    by blast
  thus TargetTerm T1\Downarrow<STCalWB SWB TWB>x
    using STCalWB-reachesBarbST
    by blast

next
case (trans P Q R R')
  assume R \to\to(\text{Calculus}(STCalWB SWB TWB)) \ast R' and R'\Downarrow<STCalWB SWB TWB>x
    and \land R'. R \to\to(\text{Calculus}(STCalWB SWB TWB)) \to R' \to R'\Downarrow<STCalWB SWB TWB>x
    \implies Q\Downarrow<STCalWB SWB TWB>x
  hence Q\Downarrow<STCalWB SWB TWB>x
  by simp
moreover assume \land Q'. Q \to\to(\text{Calculus}(STCalWB SWB TWB)) \to Q' \implies Q'\Downarrow<STCalWB SWB TWB>x
  \implies P\Downarrow<STCalWB SWB TWB>x
ultimately show $P \downarrow_{\text{STCalWB SWB TWB}} x$
by blast
qed

lemma (in encoding-wrt-barbs) enc-and-TRel-impl-indRelRTPO-respects-success:

fixes $\text{success} :: \text{barbs}$
and $\text{TRel} :: (\text{procT} \times \text{procT}) \text{ set}$

assumes $\text{encRS} :: \text{enc-respects-barb-set}\{\text{success}\}$
and $\text{trelPS} :: \text{rel-preserves-barb-set}\ \text{TRel}\ \text{TWB}\ \{\text{success}\}$
and $\text{trelRS} :: \text{rel-reflects-barb-set}\ \text{TRel}\ \text{TWB}\ \{\text{success}\}$

shows $\text{rel-respects-barb-set}(\text{indRelRTPO}\ \text{TRel})\ (\text{STCalWB SWB TWB})\ \{\text{success}\}$

proof auto

next

fix $P\ Q$

assume $P \preceq R < \text{TRel}> Q$ and $P \downarrow_{\text{STCalWB SWB TWB}} \text{success}$
thus $Q \downarrow_{\text{STCalWB SWB TWB}} \text{success}$

proof induct

case (encR $S$)

assume $\text{SourceTerm} \ S \downarrow_{\text{STCalWB SWB TWB}} \text{success}$

hence $S \downarrow_{\text{SWB}} \text{success}$
by blast

with $\text{encRS}$ have $[S] \downarrow_{\text{TWB}} \text{success}$
by simp

thus $\text{TargetTerm} ([S]) \downarrow_{\text{STCalWB SWB TWB}} \text{success}$
using $\text{STCalWB-hasBarbST}$
by blast

next

case (source $S$)

assume $\text{SourceTerm} \ S \downarrow_{\text{STCalWB SWB TWB}} \text{success}$

thus $\text{SourceTerm} \ S \downarrow_{\text{STCalWB SWB TWB}} \text{success}$
by simp

next

case (target $T1\ T2$)

assume $(T1, T2) \in \text{TRel}$

moreover assume $\text{TargetTerm} \ T1 \downarrow_{\text{STCalWB SWB TWB}} \text{success}$

hence $T1 \downarrow_{\text{TWB}} \text{success}$
using $\text{STCalWB-hasBarbST}$
by blast

ultimately have $T2 \downarrow_{\text{TWB}} \text{success}$
using $\text{trelPS}$
by simp

thus $\text{TargetTerm} \ T2 \downarrow_{\text{STCalWB SWB TWB}} \text{success}$
using $\text{STCalWB-hasBarbST}$
by blast

next

case (trans $P\ Q\ R$)

assume $P \downarrow_{\text{STCalWB SWB TWB}} \text{success}$

and $P \downarrow_{\text{STCalWB SWB TWB}} \text{success} \implies Q \downarrow_{\text{STCalWB SWB TWB}} \text{success}$

and $Q \downarrow_{\text{STCalWB SWB TWB}} \text{success} \implies R \downarrow_{\text{STCalWB SWB TWB}} \text{success}$

thus $R \downarrow_{\text{STCalWB SWB TWB}} \text{success}$
by simp

qed

next

fix $P\ Q$

assume $P \preceq R < \text{TRel}> Q$ and $Q \downarrow_{\text{STCalWB SWB TWB}} \text{success}$
thus $P \downarrow_{\text{STCalWB SWB TWB}} \text{success}$

proof induct

case (encR $S$)

assume $\text{TargetTerm} ([S]) \downarrow_{\text{STCalWB SWB TWB}} \text{success}$

hence $[S] \downarrow_{\text{TWB}} \text{success}$

242
using \text{STCalWB-hasBarbST}
by \text{blast}
with \text{encRS} have $S \downarrow \langle SWB \rangle > success$
by \text{simp}
thus \text{SourceTerm} $S \downarrow \langle STCalWB SWB TWB \rangle > success$
using \text{STCalWB-hasBarbST}
by \text{blast}

\text{next}
\text{case} (\text{source} \ S)
\text{assume} \text{SourceTerm} $S \downarrow \langle STCalWB SWB TWB \rangle > success$
thus \text{SourceTerm} $S \downarrow \langle STCalWB SWB TWB \rangle > success$
by \text{simp}

\text{next}
\text{case} (\text{target} \ T1 \ T2)
\text{assume} (T1, T2) \in \text{TRel}
moreover \text{assume} \text{TargetTerm} $T2 \downarrow \langle STCalWB SWB TWB \rangle > success$
hence $T2 \downarrow \langle TWB \rangle > success$
using \text{STCalWB-hasBarbST}
by \text{blast}
ultimately have $T1 \downarrow \langle TWB \rangle > success$
using \text{trelRS}
by \text{blast}
thus \text{TargetTerm} $T1 \downarrow \langle STCalWB SWB TWB \rangle > success$
using \text{STCalWB-hasBarbST}
by \text{blast}

\text{next}
\text{case} (\text{trans} \ P \ Q \ R)
\text{assume} R \downarrow \langle STCalWB SWB TWB \rangle > success
and $R \downarrow \langle STCalWB SWB TWB \rangle > success \implies Q \downarrow \langle STCalWB SWB TWB \rangle > success$
and $Q \downarrow \langle STCalWB SWB TWB \rangle > success \implies P \downarrow \langle STCalWB SWB TWB \rangle > success$
thus $P \downarrow \langle STCalWB SWB TWB \rangle > success$
by \text{simp}
\text{qed}

\text{qed}

\text{lemma} (\text{in} \ \text{encoding-wrt-barbs}) \ \text{enc-and-TRel-impl-indRelRTPO-respects-barbs}:
\text{fixes} \ \text{TRel} :\colon (\text{`procT} \times \text{`procT}) \text{ set}
\text{assumes} \text{encRS}: \text{enc-respects-barbs}
and \text{trelPS}: \text{rel-preserves-barbs} \ \text{TRel} \ \text{TWB}
and \text{trelRS}: \text{rel-reflects-barbs} \ \text{TRel} \ \text{TWB}
\text{shows} \text{rel-respects-barbs} \ (\text{indRelRTPO} \ \text{TRel}) \ (\text{STCalWB SWB TWB})
\text{proof} \ \text{auto}
\text{fix} \ P \ Q \ x
\text{assume} P \lessdot [\downarrow RT < \text{TRel} > Q \ and \ P \downarrow \langle STCalWB SWB TWB \rangle > x$
thus $Q\downarrow \langle STCalWB SWB TWB \rangle > x$
\text{proof} \ \text{induct}
\text{case} (\text{encR} \ S)
\text{assume} \text{SourceTerm} $S \downarrow \langle STCalWB SWB TWB \rangle > x$
hence $S \downarrow \langle SWB \rangle > x$
using \text{STCalWB-hasBarbST}
by \text{blast}
with \text{encRS} have $[S] \downarrow \langle TWB \rangle > x$
by \text{simp}
thus \text{TargetTerm} $(\langle[S]\rangle) \downarrow \langle STCalWB SWB TWB \rangle > x$
using \text{STCalWB-hasBarbST}
by \text{blast}

\text{next}
\text{case} (\text{source} \ S)
\text{assume} \text{SourceTerm} $S \downarrow \langle STCalWB SWB TWB \rangle > x$
thus \text{SourceTerm} $S \downarrow \langle STCalWB SWB TWB \rangle > x$
by \text{simp}
next case (target $T_1$, $T_2$)
assume $(T_1, T_2) \in \text{TRel}$
moreover assume TargetTerm $T_1 \downarrow <\text{STCalWB SWB TWB}>_x$
hence $T_1 \downarrow <\text{TWB}>_x$
using $\text{STCalWB-hasBarbST}$
by blast
ultimately have $T_2 \downarrow <\text{TWB}>_x$
using $\text{trelPS}$
by simp
thus TargetTerm $T_2 \downarrow <\text{STCalWB SWB TWB}>_x$
using $\text{STCalWB-hasBarbST}$
by blast

next case (trans $P$, $Q$, $R$)
assume $P \downarrow <\text{STCalWB SWB TWB}>_x$
and $P \downarrow <\text{STCalWB SWB TWB}>_x = \Rightarrow Q \downarrow <\text{STCalWB SWB TWB}>_x$
and $Q \downarrow <\text{STCalWB SWB TWB}>_x = \Rightarrow R \downarrow <\text{STCalWB SWB TWB}>_x$
thus $R \downarrow <\text{STCalWB SWB TWB}>_x$
by simp
qed

next

fix $P$, $Q$, $x$
assume $P \lessapprox [\llbracket RT \rrbracket \downarrow <\text{TRel}> Q$ and $Q \downarrow <\text{STCalWB SWB TWB}>_x$
thus $P \downarrow <\text{STCalWB SWB TWB}>_x$

proof induct

case (encR $S$)
assume TargetTerm $([S]) \downarrow <\text{STCalWB SWB TWB}>_x$
hence $[S] \downarrow <\text{TWB}>_x$
using $\text{STCalWB-hasBarbST}$
by blast
with encRS have $S \downarrow <\text{SWB}>_x$
by simp
thus SourceTerm $S \downarrow <\text{STCalWB SWB TWB}>_x$
using $\text{STCalWB-hasBarbST}$
by blast

next case (source $S$)
assume SourceTerm $S \downarrow <\text{STCalWB SWB TWB}>_x$
thus SourceTerm $S \downarrow <\text{STCalWB SWB TWB}>_x$
by simp

next
case (target $T_1$, $T_2$)
assume $(T_1, T_2) \in \text{TRel}$
moreover assume TargetTerm $T_2 \downarrow <\text{STCalWB SWB TWB}>_x$
hence $T_2 \downarrow <\text{TWB}>_x$
using $\text{STCalWB-hasBarbST}$
by blast
ultimately have $T_1 \downarrow <\text{TWB}>_x$
using $\text{trelRS}$
by blast
thus TargetTerm $T_1 \downarrow <\text{STCalWB SWB TWB}>_x$
using $\text{STCalWB-hasBarbST}$
by blast

next case (trans $P$, $Q$, $R$)
assume $R \downarrow <\text{STCalWB SWB TWB}>_x$
and $R \downarrow <\text{STCalWB SWB TWB}>_x = \Rightarrow Q \downarrow <\text{STCalWB SWB TWB}>_x$
and $Q \downarrow <\text{STCalWB SWB TWB}>_x = \Rightarrow P \downarrow <\text{STCalWB SWB TWB}>_x$
thus $P \downarrow <\text{STCalWB SWB TWB}>_x$
by simp

244
An encoding is success sensitive and operational corresponding w.r.t. a bisimulation TRel that respects success iff there exists a bisimulation that includes TRel and respects success. The same holds if we consider not only success sensitiveness but barb sensitiveness in general.

**Lemma** (in `encoding-wrt-barbs`) OC-SS-iff-source-target-rel:

```plaintext
proof
assumes B1: rel-weakly-preserves-barb-set TRel TWB {success}
  and B2: rel-weakly-reflects-barb-set TRel TWB {success}
  and B3: enc-weakly-preserves-barb-set {success}
  and B4: enc-weakly-reflects-barb-set {success}

define Rel where Rel = indRelRTPO TRel

hence B1: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
by (simp add: indRelRTPO.encR)

defined Rel-def have B2: ∀ T1 T2. (T1, T2) ∈ TRel (TargetTerm T1, TargetTerm T2) ∈ Rel
by (simp add: indRelRTPO.target)

defined Rel-def have B3: ∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel (T1, T2) ∈ TRel+
by (simp add: indRelRTPO-to-TRel(2)[where TRel=TRel]
  trans-closure-of-TRel-refl-cond[where TRel=TRel]

apply (rule ifI, (erule conjE)+)

uses B1 B2 B3 B4 B5 B6 by blast

next
assumes B1: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
  ∧ (∀ T1 T2. (T1, T2) ∈ TRel (TargetTerm T1, TargetTerm T2) ∈ Rel)
  ∧ (∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel (T1, T2) ∈ TRel+)
  ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel ([S], T) ∈ TRel+)
  ∧ weak-reduction-bisimulation Rel (STCal Source Target)
  ∧ rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) {success}
```

Apply (rule exI) using B1 B2 B3 B4 B5 B6 by blast
\[ \land \text{rel-weakly-respects-barb-set } \text{Rel} \; (\text{STCalWB SWB TWB}) \; \{ \text{success} \} \]

**from** this obtain **Rel** where \(C1: \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel} \)
and \(C2: \forall T1 T2. (T1, T2) \in \text{TRel} \to (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel} \)
and \(C3: \forall T1 T2. (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel} \to (T1, T2) \in \text{TRel}^+ \)
and \(C4: \forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \to ([S], T) \in \text{TRel}^* \)
and \(C5: \text{weak-reduction-bisimulation } \text{Rel} \; (\text{STCal Source Target}) \)
and \(C6: \text{weak-reduction-bisimulation } \text{Rel} \; (\text{STCalWB SWB TWB}) \; \{ \text{success} \} \)

by **auto**

from \(C1 \; C2 \; C3 \; C4 \; C5 \) have \(\exists \text{Rel}. (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}) \land (\forall T1 T2. (T1, T2) \in \text{TRel} \to (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}) \land (\forall T1 T2. (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel} \to (T1, T2) \in \text{TRel}^+) \land (\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \to ([S], T) \in \text{TRel}^*) \land \text{weak-reduction-bisimulation } \text{Rel} \; (\text{STCal Source Target}) \)

by **blast**

**hence** **operational-corresponding** (\( \text{TRel}^+ \))

\[ \land \text{weak-reduction-bisimulation } (\text{TRel}^+) \; \text{Target} \]

**using** \(OC\text{-iff-weak-reduction-bisimulation}[\text{where } \text{TRel}=\text{TRel}] \)

by **auto**

moreover have \(\exists \text{Rel}. (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}) \land \text{rel-weakly-respects-barb-set } \text{Rel} \; (\text{STCalWB SWB TWB}) \; \{ \text{success} \} \)

**apply** (rule **exI**) **using** \(C1 \; C6\) by **blast**

**hence** **enc-weakly-respects-barb-set** \{ **success** \}

**using** **success-sensitive-iff-source-target-rel-weakly-respects-success**

by **auto**

moreover have **rel-weakly-respects-barb-set** \( \text{TRel} \; \text{TWB} \; \{ \text{success} \} \)

**proof** **auto**

fix \( TP \; TQ \; TP' \)

**assume** \((TP, TQ) \in \text{TRel})

with **C2 have** \((\text{TargetTerm } TP, \text{TargetTerm } TQ) \in \text{Rel})

by **simp**

moreover assume \( TP \to (\text{Calculus TWB})^* \; TP' \) and \( TP'^{<TWB>}\; \text{success} \)

**hence** \( \text{TargetTerm } TP'^{<STCalWB SWB TWB>success} \)

**using** \(\text{STCalWB-reachesBarbST} \)

by **blast**

ultimately have \( \text{TargetTerm } TP'^{<STCalWB SWB TWB>success} \)

**using** \(C6\)

by **blast**

thus \( TQ'^{<TWB>}\; \text{success} \)

**using** \(\text{STCalWB-reachesBarbST} \)

by **blast**

next

fix \( TP \; TQ \; TQ' \)

**assume** \((TP, TQ) \in \text{TRel})

with **C2 have** \((\text{TargetTerm } TP, \text{TargetTerm } TQ) \in \text{Rel})

by **simp**

moreover assume \( TQ \to (\text{Calculus TWB})^* \; TQ' \) and \( TQ'^{<TWB>}\; \text{success} \)

**hence** \( \text{TargetTerm } TQ'^{<STCalWB SWB TWB>success} \)

**using** \(\text{STCalWB-reachesBarbST} \)

by **blast**

ultimately have \( \text{TargetTerm } TQ'^{<STCalWB SWB TWB>success} \)

**using** \(C6\)

by **blast**

thus \( TP'^{<TWB>}\; \text{success} \)

**using** \(\text{STCalWB-reachesBarbST} \)

by **blast**

qed

ultimately show **operational-corresponding** (\( \text{TRel}^+ \))

\[ \land \text{weak-reduction-bisimulation } (\text{TRel}^+) \; \text{Target} \]

\[ \land \text{enc-weakly-respects-barb-set} \; \{ \text{success} \} \land \text{rel-weakly-respects-barb-set } \text{TRel} \; \text{TWB} \; \{ \text{success} \} \]

by **fast**

qed
lemma (in encoding-wrt-barbs) OC-SS-RB-iff-source-target-rel:
  fixes success :: 'bars
  and TRel :: ('proc T × 'proc T) set
  shows (operational-corresponding (TRel')
         ∧ weak-reduction-bisimulation (TRel') Target
         ∧ enc-weakly-respects-bars ∧ enc-weakly-respects-barb-set {success}
         ∧ rel-weakly-respects-bars TRel TRWB ∧ rel-weakly-respects-barb-set TRel TRWB {success})
  = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
      ∧ (∀ T1 T2. (T1, T2) ∈ TRel → (TargetTerm T1, TargetTerm T2) ∈ Rel)
      ∧ (∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel → (T1, T2) ∈ TRel+)
      ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel*)
      ∧ weak-reduction-bisimulation Rel (STCal Source Target)
      ∧ rel-weakly-respects-bars Rel (STCalWB SWB WB) ∧
      rel-weakly-respects-barb-set Rel (STCalWB SWB WB) {success})
proof (rule iffI, (erule conjE)+)
  assume A1: rel-weakly-preserves-barb-set TRel TRWB {success}
  and A2: rel-weakly-reflects-barb-set TRel TRWB {success}
  and A3: enc-weakly-preserves-barb-set {success}
  and A4: enc-weakly-reflects-barb-set {success}
  and A5: rel-weakly-preserves-bars TRel TRWB and A6: rel-weakly-reflects-bars TRel TRWB
  and A7: enc-weakly-preserves-bars and A8: enc-weakly-reflects-bars

  define Rel where Rel = indRelRTPO TRel
  hence B1: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
    by (simp add: indRelRTPO.encR)
  from Rel-def have B2: ∀ T1 T2. (T1, T2) ∈ TRel → (TargetTerm T1, TargetTerm T2) ∈ Rel
    by (simp add: indRelRTPO.encR)
  from Rel-def have B3: ∀ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ Rel → (T1, T2) ∈ TRel+
    by (simp add: indRelRTPO-to-TRel(4)[where TRel=TRel])
  from Rel-def have B4: ∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel*
    using indRelRTPO-to-TRel(2)[where TRel=TRel]
      trans-closure-of-TRel-refl-cond[where TRel=TRel]
    by simp
  assume A1 A2 A3 A4 have B6: rel-weakly-respects-barb-set Rel (STCalWB SWB WB) {success}
    using enc-and-TRel-impl-indRelRTPO-weakly-respects-success[where TRel=TRel]
    and success=success
    by blast
  from Rel-def A5 A6 A7 A8 have B7: rel-weakly-respects-bars Rel (STCalWB SWB WB) {success}
    using enc-and-TRel-impl-indRelRTPO-weakly-respects-bars[where TRel=TRel]
  by blast
next
  assume A1 A2 A3 A4 A5 A6 A7 A8 have B7: rel-weakly-respects-barb-set Rel (STCalWB SWB WB) {success}
  by blast
show (rel-weakly-preserves-barb-set Rel (STCalWB SWB WB)) {success}
  apply (rule exI) using B1 B2 B3 B4 B5 B6 B7 by blast
\textit{TRel} \textit{TRel} \textit{TWB} \textit{Rel} \textit{StCalWB} \textit{SWB} \textit{TBW} \\
\textit{this obtain} \textit{Rel} \textit{where} \textit{C} (\forall \textit{S}. (\textit{SourceTerm} \textit{S}, \textit{TargetTerm} ([\textit{S}])) \in \textit{Rel}) \\
\land (\forall \textit{T1} \textit{T2}. (\textit{T1}, \textit{T2}) \in \textit{Rel} \rightarrow (\textit{TargetTerm} \textit{T1}, \textit{TargetTerm} \textit{T2}) \in \textit{Rel}) \\
\land (\forall \textit{T1} \textit{T2}. (\textit{TargetTerm} \textit{T1}, \textit{TargetTerm} \textit{T2}) \in \textit{Rel} \rightarrow (\textit{T1}, \textit{T2}) \in \textit{TRel}^+) \\
\land (\forall \textit{S} \textit{T}. (\textit{SourceTerm} \textit{S}, \textit{TargetTerm} \textit{T}) \in \textit{Rel} \rightarrow ([\textit{S}], \textit{T}) \in \textit{TRel}^+) \\
\land \textit{weak-reduction-bisimulation} \textit{Rel} \textit{(StCal Source Target)} \\
\land \textit{rel-weakly-respects-barbs} \textit{Rel} \textit{(StCalWB SWB TBW)} \\
\land \textit{rel-weakly-respects-barb-set} \textit{Rel} \textit{(StCalWB SWB TBW)} \{ \textit{success} \} \\
\textit{by auto} \\
\textit{hence} \textit{C1} (\forall \textit{S}. (\textit{SourceTerm} \textit{S}, \textit{TargetTerm} ([\textit{S}])) \in \textit{Rel}) \\
\textit{by simp} \\
\textit{from} \textit{C} \textit{have} \textit{C2} (\forall \textit{T1} \textit{T2}. (\textit{T1}, \textit{T2}) \in \textit{Rel} \rightarrow (\textit{TargetTerm} \textit{T1}, \textit{TargetTerm} \textit{T2}) \in \textit{Rel}) \\
\textit{by simp} \\
\textit{from} \textit{C} \textit{have} \textit{C3} (\forall \textit{T1} \textit{T2}. (\textit{TargetTerm} \textit{T1}, \textit{TargetTerm} \textit{T2}) \in \textit{Rel} \rightarrow (\textit{T1}, \textit{T2}) \in \textit{TRel}^+) \\
\textit{by simp} \\
\textit{from} \textit{C} \textit{have} \textit{C4} (\forall \textit{S} \textit{T}. (\textit{SourceTerm} \textit{S}, \textit{TargetTerm} \textit{T}) \in \textit{Rel} \rightarrow ([\textit{S}], \textit{T}) \in \textit{TRel}^+) \\
\textit{by simp} \\
\textit{from} \textit{C} \textit{have} \textit{C5} \textit{weak-reduction-bisimulation} \textit{Rel} \textit{(StCal Source Target)} \\
\textit{by simp} \\
\textit{from} \textit{C} \textit{have} \textit{C6} \textit{rel-weakly-respects-barbs} \textit{Rel} \textit{(StCalWB SWB TBW)} \\
\textit{apply (rule conjE)} \textit{apply (erule conjE)+ by blast} \\
\textit{from} \textit{C} \textit{have} \textit{C7} \textit{rel-weakly-respects-barb-set} \textit{Rel} \textit{(StCalWB SWB TBW)} \{ \textit{success} \} \\
\textit{apply (rule conjE)} \textit{apply (erule conjE)+ by blast} \\
\textit{from} \textit{C1} \textit{C2} \textit{C3} \textit{C4} \textit{C5} \textit{have} \exists \textit{Rel}. (\forall \textit{S}. (\textit{SourceTerm} \textit{S}, \textit{TargetTerm} ([\textit{S}])) \in \textit{Rel}) \\
\land (\forall \textit{T1} \textit{T2}. (\textit{T1}, \textit{T2}) \in \textit{Rel} \rightarrow (\textit{TargetTerm} \textit{T1}, \textit{TargetTerm} \textit{T2}) \in \textit{Rel}) \\
\land (\forall \textit{T1} \textit{T2}. (\textit{TargetTerm} \textit{T1}, \textit{TargetTerm} \textit{T2}) \in \textit{Rel} \rightarrow (\textit{T1}, \textit{T2}) \in \textit{TRel}^+) \\
\land (\forall \textit{S} \textit{T}. (\textit{SourceTerm} \textit{S}, \textit{TargetTerm} \textit{T}) \in \textit{Rel} \rightarrow ([\textit{S}], \textit{T}) \in \textit{TRel}^+) \\
\land \textit{weak-reduction-bisimulation} \textit{Rel} \textit{(StCal Source Target)} \\
\textit{by blast} \\
\textit{hence} \textit{operational-corresponding} (\textit{TRel}^+) \\
\land \textit{weak-reduction-bisimulation} (\textit{TRel}^+) \textit{Target} \\
\textit{using OC-iff-weak-reduction-bisimulation[where TRel=TRel]} \\
\textit{by auto} \\
\textit{moreover have} \exists \textit{Rel}. (\forall \textit{S}. (\textit{SourceTerm} \textit{S}, \textit{TargetTerm} ([\textit{S}])) \in \textit{Rel}) \\
\land \textit{rel-weakly-respects-barb-set} \textit{Rel} \textit{(StCalWB SWB TBW)} \{ \textit{success} \} \\
\textit{apply (rule exI)} \textit{using C1 C6 by blast} \\
\textit{hence} \textit{enc-weakly-respects-barb-set} \{ \textit{success} \} \\
\textit{using success-sensitive-iff-source-target-rel-weakly-respects-success} \\
\textit{by auto} \\
\textit{moreover have} \exists \textit{Rel}. (\forall \textit{S}. (\textit{SourceTerm} \textit{S}, \textit{TargetTerm} ([\textit{S}])) \in \textit{Rel}) \\
\land \textit{rel-weakly-respects-barbs} \textit{Rel} \textit{(StCalWB SWB TBW)} \\
\textit{apply (rule exI)} \textit{using C1 C7 by blast} \\
\textit{hence} \textit{enc-weakly-respects-barbs} \\
\textit{using enc-weakly-respects-barbs-iff-source-target-rel} \\
\textit{by auto} \\
\textit{moreover have} \textit{rel-weakly-respects-barb-set} \textit{TRel} \textit{TBW} \{ \textit{success} \} \\
\textit{proof auto} \\
\textit{fix} \textit{TP} \textit{TQ} \textit{TP'} \\
\textit{assume} (\textit{TP}, \textit{TQ}) \in \textit{TRel} \\
\textit{with} \textit{C2} \textit{have} (\textit{TargetTerm} \textit{TP}, \textit{TargetTerm} \textit{TQ}) \in \textit{Rel} \\
\textit{by simp} \\
\textit{moreover assume} \textit{TP} \rightarrow (\textit{Calculus TBW})* \textit{TP'} \textit{and TP'}<\textit{TWB}>\textit{success} \\
\textit{hence} \textit{TargetTerm} \textit{TP} \downarrow<\textit{STCalWB SWB TBW}>\textit{success} \\
\textit{using STCalWB-reachesBarbST} \\
\textit{by blast} \\
\textit{ultimately have} \textit{TargetTerm} \textit{TQ} \downarrow<\textit{STCalWB SWB TBW}>\textit{success} \\
\textit{using C6} \\
\textit{by blast} \\
\textit{thus} \textit{TQ} \downarrow<\textit{TWB}>\textit{success} \\
\textit{using STCalWB-reachesBarbST}
by blast

next

fix $TP$, $TQ$, $TQ'$
assume $(TP, TQ) \in TRel$
with $C2$ have $(\text{TargetTerm } TP, \text{TargetTerm } TQ) \in \text{Rel}$
by simp

moreover assume $TQ \mapsto (\text{Calculus TWB})^* TQ'$ and $TQ' \downarrow <\text{TWB}> success$
hence $\text{TargetTerm } TQ \downarrow <\text{STCalWB} \ \text{SWB} \ TWB>_x success$
using $\text{STCalWB-reachesBarbST}$
by blast

ultimately have $\text{TargetTerm } TP \downarrow <\text{STCalWB} \ \text{SWB} \ TWB>_x success$
using $C6$
by blast

thus $TP \downarrow <\text{TWB}> x$
using $\text{STCalWB-reachesBarbST}$
by blast

qed

moreover have rel-weakly-respects-barbs $TRel$ $\text{TWB}$

proof auto

fix $TP$, $TQ$, $x$, $TP'$
assume $(TP, TQ) \in TRel$
with $C2$ have $(\text{TargetTerm } TP, \text{TargetTerm } TQ) \in \text{Rel}$
by simp

moreover assume $TP \mapsto (\text{Calculus TWB})^* TP'$ and $TP' \downarrow <\text{TWB}> x$
hence $\text{TargetTerm } TP \downarrow <\text{STCalWB} \ \text{SWB} \ TWB>_x$
using $\text{STCalWB-reachesBarbST}$
by blast

ultimately have $\text{TargetTerm } TQ \downarrow <\text{STCalWB} \ \text{SWB} \ TWB>_x$
using $C7$
by blast

thus $TQ \downarrow <\text{TWB}> x$
using $\text{STCalWB-reachesBarbST}$
by blast

next

fix $TP$, $TQ$, $TQ'$
assume $(TP, TQ) \in TRel$
with $C2$ have $(\text{TargetTerm } TP, \text{TargetTerm } TQ) \in \text{Rel}$
by simp

moreover assume $TQ \mapsto (\text{Calculus TWB})^* TQ'$ and $TQ' \downarrow <\text{TWB}> x$
hence $\text{TargetTerm } TQ \downarrow <\text{STCalWB} \ \text{SWB} \ TWB>_x$
using $\text{STCalWB-reachesBarbST}$
by blast

ultimately have $\text{TargetTerm } TP \downarrow <\text{STCalWB} \ \text{SWB} \ TWB>_x$
using $C7$
by blast

thus $TP \downarrow <\text{TWB}> x$
using $\text{STCalWB-reachesBarbST}$
by blast

qed

ultimately show operational-corresponding ($TRel$*)
\land weak-reduction-bisimulation ($TRel^*$) Target
\land enc-weakly-respects-barbs \ \text{enc-weakly-respects-barb-set} \ {\text{success}}
\land rel-weakly-respects-barbs $TRel$ $\text{TWB}$ \ \text{rel-weakly-respects-barb-set} $TRel$ $\text{TWB}$ \ {\text{success}}
by fast

qed

lemma (in encoding-wrt-barbs) OC-SS-wrt-preorder-iff-source-target-rel:

fixes success :: 'barbs
and $TRel$ :: ('procT $\times$ 'procT) set
shows (operational-corresponding $TRel$ \ \text{preorder} $TRel$ \ \text{weak-reduction-bisimulation} $TRel$ Target
\land enc-weakly-respects-barb-set \ {\text{success}})
\begin{align*}
\wedge \text{rel-weakly-respects-barb-set TRel TWB \{success\}} \\
= (\exists \text{Rel}. \ (\forall S. \ (\text{SourceTerm} S, \ \text{TargetTerm} ([S])) \in \text{Rel}) \\
\wedge \text{TRel} = \{(T1, T2), \ (\text{TargetTerm} T1, \ \text{TargetTerm} T2) \in \text{Rel}\} \\
\wedge (\forall S T. \ (\text{SourceTerm} S, \ \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}) \\
\wedge \text{weak-reduction-bisimulation Rel (STCal Source Target) \wedge preorder Rel} \\
\wedge \text{rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) \{success\}}
\end{align*}

\textbf{proof (rule iffI, (erule conjI)+)}

\textbf{assume A1:} rel-weakly-preserves-barb-set TRel TWB \{success\} \\
\textbf{and A2:} rel-weakly-reflects-barb-set TRel TWB \{success\} \\
\textbf{and A3: enc-weakly-preserves-barb-set \{success\}} \\
\textbf{and A4: enc-weakly-reflects-barb-set \{success\}} \\
\textbf{and A5:} preorder TRel

\textbf{from A5 have A6: TRel* = TRel} \\
\textbf{using trancl-id[of TRel] preorder-on-def} \\
\textbf{by blast}

\textbf{from A5 have A7: TRel* = TRel} \\
\textbf{using reflcl-trancl[of TRel] trancl-id[of TRel]} \\
\textbf{unfolding refl-on-def preorder-on-def} \\
\textbf{by auto}

\textbf{define Rel where Rel = indRelRTPO TRel}

\textbf{hence B1:} \ \forall S. \ (\text{SourceTerm} S, \ \text{TargetTerm} ([S])) \in \text{Rel}
\textbf{by (simp add: indRelRTPO.\_encR)}

\textbf{from Rel-def A6 have B2:} \text{TRel} = \{(T1, T2), \ (\text{TargetTerm} T1, \ \text{TargetTerm} T2) \in \text{Rel}\}
\textbf{by (auto simp add: indRelRTPO.\_target)}

\textbf{from Rel-def A7 have B3:} \ \forall S T. \ (\text{SourceTerm} S, \ \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}
\textbf{by simp}

\textbf{assume operational-complete TRel and operational-sound TRel} \\
\textbf{and weak-reduction-simulation TRel Target} \\
\textbf{and} \ \forall P Q Q'. \ (P, Q) \in \text{TRel} \wedge Q \rightarrow \text{Target*} \ Q' \rightarrow (\exists P', P \rightarrow \text{Target*} \ P' \wedge (P', Q') \in \text{TRel})

\textbf{with Rel-def A6 A7 have B4: weak-reduction-bisimulation Rel (STCal Source Target)}
\textbf{by simp}

\textbf{from Rel-def A5 have B5: preorder Rel} \\
\textbf{using indRelRTPO-is-preorder[where TRel=TRel]} \\
\textbf{unfolding preorder-on-def} \\
\textbf{by blast}

\textbf{from Rel-def A1 A2 A3 A4 have B6: rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) \{success\}} \\
\textbf{using enc-and-TRel-impl-indRelRTPO-weakly-respects-success[where TRel=TRel]} \\
\textbf{and success=succ} \\
\textbf{by blast}

\textbf{show} \exists \text{Rel}. \ (\forall S. \ (\text{SourceTerm} S, \ \text{TargetTerm} ([S])) \in \text{Rel}) \\
\wedge \text{TRel} = \{(T1, T2), \ (\text{TargetTerm} T1, \ \text{TargetTerm} T2) \in \text{Rel}\} \\
\wedge (\forall S T. \ (\text{SourceTerm} S, \ \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}) \\
\wedge \text{weak-reduction-bisimulation Rel (STCal Source Target) \wedge preorder Rel} \\
\wedge \text{rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) \{success\}}
\textbf{apply (rule exI) using B1 B2 B3 B4 B5 B6 by blast}

\textbf{next}
\textbf{assume} \exists \text{Rel}. \ (\forall S. \ (\text{SourceTerm} S, \ \text{TargetTerm} ([S])) \in \text{Rel}) \\
\wedge \text{TRel} = \{(T1, T2), \ (\text{TargetTerm} T1, \ \text{TargetTerm} T2) \in \text{Rel}\} \\
\wedge (\forall S T. \ (\text{SourceTerm} S, \ \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}) \\
\wedge \text{weak-reduction-bisimulation Rel (STCal Source Target) \wedge preorder Rel} \\
\wedge \text{rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) \{success\}}
\textbf{from this obtain Rel where C1:} \ (\forall S. \ (\text{SourceTerm} S, \ \text{TargetTerm} ([S])) \in \text{Rel}) \\
\textbf{and C2: TRel} = \{(T1, T2), \ (\text{TargetTerm} T1, \ \text{TargetTerm} T2) \in \text{Rel}\} \\
\textbf{and C3:} \ (\forall S T. \ (\text{SourceTerm} S, \ \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}) \\
\textbf{and C4: weak-reduction-bisimulation Rel (STCal Source Target) and C5: preorder Rel} \\
\textbf{and C6: rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) \{success\}} \\
\textbf{by auto}

250
from $C_1$ $C_2$ $C_3$ $C_4$ $C_5$ have $\exists \text{Rel} \forall S. (\text{SourceTerm } S, \text{ TargetTerm } ([S])) \in \text{Rel}$

$\wedge (\text{Rel} = \{(T_1, T_2). (\text{TargetTerm } T_1, \text{TargetTerm } T_2) \in \text{Rel}\})$

$\wedge \forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel} \wedge \text{preorder} \text{Rel}$

$\wedge \text{weak-reduction-bisimulation} \text{Rel} (\text{STCal Source Target})$

by blast

hence operational-corresponding $\text{TRel} \wedge \text{preorder} \text{TRel} \wedge \text{weak-reduction-bisimulation} \text{TRel} \text{Target}$

using OC-wrt-preorder-iff-weak-reduction-bisimulation[where $\text{TRel}=\text{TRel}]

by simp

moreover have $\exists \text{Rel}. \forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}$

$\wedge \text{rel-weakly-respects-barb-set} \text{Rel}$

apply (rule exI) using $C_1 C_6$ by blast

hence $\text{TRel} \rightarrow\!
\!\!\!
\left(\begin{array}{c}
\text{success}
\end{array}\right)$

using success-sensitive-iff-source-target-rel-weakly-respects-success

by simp

moreover have $\text{rel-weakly-respects-barb-set} \text{TRel} \text{TWB} \{ \text{success} \}$

proof auto

fix $TP TQ TP'$

assume $(TP, TQ) \in \text{TRel}$

with $C_2$ have $(\text{TargetTerm } TP, \text{TargetTerm } TQ) \in \text{Rel}$

by simp

moreover assume $TP \rightarrow (\text{Calculus TWB})^{*} TP'$ and $TP' \downarrow \left< \text{TWB} \right> \text{success}$

hence $\text{TargetTerm } TP \downarrow \left< \text{STCalWB SWB TWB} \right> \text{success}$

using $\text{STCalWB-reachesBarbST}$

by blast

ultimately have $\text{TargetTerm } TQ \downarrow \left< \text{STCalWB SWB TWB} \right> \text{success}$

using $C_6$

by blast

thus $TP \downarrow \left< \text{TWB} \right> \text{success}$

using $\text{STCalWB-reachesBarbST}$

by blast

next

fix $TP TQ TQ'$

assume $(TP, TQ) \in \text{TRel}$

with $C_2$ have $(\text{TargetTerm } TP, \text{TargetTerm } TQ) \in \text{Rel}$

by simp

moreover assume $TQ \rightarrow (\text{Calculus TWB})^{*} TQ'$ and $TQ' \downarrow \left< \text{TWB} \right> \text{success}$

hence $\text{TargetTerm } TQ \downarrow \left< \text{STCalWB SWB TWB} \right> \text{success}$

using $\text{STCalWB-reachesBarbST}$

by blast

ultimately have $\text{TargetTerm } TP \downarrow \left< \text{STCalWB SWB TWB} \right> \text{success}$

using $C_6$

by blast

thus $TP \downarrow \left< \text{TWB} \right> \text{success}$

using $\text{STCalWB-reachesBarbST}$

by blast

qed

ultimately show operational-corresponding $\text{TRel} \wedge \text{preorder} \text{TRel}$

$\wedge \text{weak-reduction-bisimulation} \text{TRel} \text{Target}$

$\wedge \text{enc-weakly-respects-barb-set} \{ \text{success} \} \wedge \text{rel-weakly-respects-barb-set} \text{TRel} \text{TWB} \{ \text{success} \}$

by fast

qed

lemma (in encoding-wrt-barbs) OC-SS-RR-wrt-preorder-iff-source-target-rel:

fixes $\text{success} :: \text{`barbs}$

and $\text{TRel} :: (\text{`procT} \times \text{`procT}) \text{ set}$

shows (operational-corresponding $\text{TRel} \wedge \text{preorder} \text{TRel} \wedge \text{weak-reduction-bisimulation} \text{TRel} \text{Target}$

$\wedge \text{enc-weakly-respects-barbs} \wedge \text{rel-weakly-respects-barbs} \text{TRel} \text{TWB}$

$\wedge \text{enc-weakly-respects-barb-set} \{ \text{success} \}$

$\wedge \text{rel-weakly-respects-barb-set} \text{TRel} \text{TWB} \{ \text{success} \}$

$= (\exists \text{Rel}. (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel})$

$\wedge \text{TRel} = \{(T_1, T_2). (\text{TargetTerm } T_1, \text{TargetTerm } T_2) \in \text{Rel}\}$
\[ \forall S \ T. \ (\text{SourceTerm } S, \ \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], \ T) \in \text{TRel} \]
\[ \land \ \text{weak-reduction-bisimulation } \text{Rel} \ (\text{STCal Source Target}) \land \ \text{preorder } \text{Rel} \]
\[ \land \ \text{rel-weakly-respects-barbs } \text{Rel} \ (\text{STCalWB SWB TWB}) \]
\[ \land \ \text{rel-weakly-respects-barb-set } \text{Rel} \ (\text{STCalWB SWB TWB}) \ \{ \text{success} \} \]

**proof** (rule \text{iff}, (\text{rule conj})+)

**assume** \(A1: \text{rel-weakly-preserves-barbs } \text{TRel} \text{ TWB} \) \text{ and } \(A2: \text{rel-weakly-reflects-barbs } \text{TRel} \text{ TWB} \)

\[ \land \ A3: \text{enc-weakly-preserves-barbs} \text{ and } \ A4: \text{enc-weakly-reflects-barbs} \]

\[ \land \ A5: \text{preorder } \text{TRel} \]

**from** \(A5 \ \text{have} A6: \text{TRel}^+ = \text{TRel} \)

**using** \(\text{transl-id[of TRel]}\)

**unfolding** \(\text{preorder-on-def} \)

**by** \(\text{blast} \)

**from** \(A5 \ \text{have} A7: \text{TRel}^+ = \text{TRel} \)

**using** \(\text{refl-transl[of TRel]} \)

**unfolding** \(\text{preorder-on-def refl-on-def} \)

**by** \(\text{auto} \)

**define** \(\text{Rel where} \)

\[ \text{Rel} = \text{indRelRTPO} \text{TRel} \]

**hence** \(B1: \forall S. \ (\text{SourceTerm } S, \ \text{TargetTerm } ([S])) \in \text{Rel} \)

**by** \((\text{simp add: indRelRTPO.encR}) \)

**from** \(\text{Rel-def A6 have B2: TRel} = \{(T1, \ T2). \ (\text{TargetTerm } T1, \ \text{TargetTerm } T2) \in \text{Rel} \}

**using** \(\text{indRelRTPO-to-TRel(4)[where TRel=TRel]} \)

**by** \((\text{auto simp add: indRelRTPO.target}) \)

**from** \(\text{Rel-def A7 have B3: } \forall S \ T. \ (\text{SourceTerm } S, \ \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], \ T) \in \text{TRel} \)

**using** \(\text{indRelRTPO-to-TRel(2)[where TRel=TRel]} \)

\[ \text{trans-closure-of-TRel-refl-cond[where TRel=TRel]} \]

**by** \(\text{simp} \)

**assume** \(\text{operational-complete TRel} \text{ and } \text{operational-sound TRel} \)

\[ \land \ \text{weak-reduction-simulation } \text{TRel Target} \]

\[ \land \ \forall P \ Q \ Q'. \ (P, \ Q) \in \text{TRel} \land \ Q \rightarrow \text{Target} \rightarrow Q' \rightarrow (\exists P'. \ P \rightarrow \text{Target} \rightarrow P' \land (P', \ Q') \in \text{TRel}) \]

**with** \(\text{Rel-def A6 A7 have B4: weak-reduction-bisimulation } \text{Rel} \ (\text{STCal Source Target}) \)

**using** \(\text{OC-iff-indRelRTPO-is-weak-reduction-bisimulation[where TRel=TRel]} \)

**by** \(\text{simp} \)

**from** \(\text{Rel-def A5 have B5: preorder } \text{Rel} \)

**using** \(\text{indRelRTPO-is-preorder[where TRel=TRel]} \)

**unfolding** \(\text{preorder-on-def} \)

**by** \(\text{blast} \)

**from** \(\text{Rel-def A1 A2 A3 A4 have B6: rel-weakly-respects-barbs } \text{Rel} \ (\text{STCalWB SWB TWB}) \)

**using** \(\text{enc-and-TRel-impl-indRelRTPO-weakly-respects-barbs[where TRel=TRel]} \)

**by** \(\text{blast} \)

**hence** \(B7: \text{rel-weakly-respects-barb-set } \text{Rel} \ (\text{STCalWB SWB TWB}) \ \{ \text{success} \} \)

**by** \(\text{blast} \)

**show** \(\exists \text{Rel. } \forall S. \ (\text{SourceTerm } S, \ \text{TargetTerm } ([S])) \in \text{Rel} \)

\[ \land \ \text{TRel} = \{(T1, \ T2). \ (\text{TargetTerm } T1, \ \text{TargetTerm } T2) \in \text{Rel} \}

\[ \land \ (\forall S \ T. \ (\text{SourceTerm } S, \ \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], \ T) \in \text{TRel} \)

\[ \land \ \text{weak-reduction-bisimulation } \text{Rel} \ (\text{STCal Source Target}) \land \ \text{preorder } \text{Rel} \]

\[ \land \ \text{rel-weakly-respects-barbs } \text{Rel} \ (\text{STCalWB SWB TWB}) \]

\[ \land \ \text{rel-weakly-respects-barb-set } \text{Rel} \ (\text{STCalWB SWB TWB}) \ \{ \text{success} \} \]

**apply** \((\text{rule exI}) \) **using** \(B1 \ B2 \ B3 \ B4 \ B5 \ B6 \ B7 \) **by** \(\text{blast} \)

**next**

**assume** \(\exists \text{Rel. } \forall S. \ (\text{SourceTerm } S, \ \text{TargetTerm } ([S])) \in \text{Rel} \)

\[ \land \ \text{TRel} = \{(T1, \ T2). \ (\text{TargetTerm } T1, \ \text{TargetTerm } T2) \in \text{Rel} \}

\[ \land \ (\forall S \ T. \ (\text{SourceTerm } S, \ \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], \ T) \in \text{TRel} \)

\[ \land \ \text{weak-reduction-bisimulation } \text{Rel} \ (\text{STCal Source Target}) \land \ \text{preorder } \text{Rel} \]

\[ \land \ \text{rel-weakly-respects-barbs } \text{Rel} \ (\text{STCalWB SWB TWB}) \]

\[ \land \ \text{rel-weakly-respects-barb-set } \text{Rel} \ (\text{STCalWB SWB TWB}) \ \{ \text{success} \} \]

**from** \((\text{this obtain Rel where}) \)

**C1: \ (\forall S. \ (\text{SourceTerm } S, \ \text{TargetTerm } ([S])) \in \text{Rel}) \)

**and** \(\text{C2: TRel} = \{(T1, \ T2). \ (\text{TargetTerm } T1, \ \text{TargetTerm } T2) \in \text{Rel} \}

**and** \(\text{C3: } (\forall S \ T. \ (\text{SourceTerm } S, \ \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], \ T) \in \text{TRel} \)

**and** \(\text{C4: weak-reduction-bisimulation } \text{Rel} \ (\text{STCal Source Target}) \text{ and } \ C5: \text{preorder } \text{Rel} \)

**and** \(\text{C6: rel-weakly-respects-barbs } \text{Rel} \ (\text{STCalWB SWB TWB}) \)

**by** \(\text{auto} \)
from C1 C2 C3 C4 C5 have \( \exists \text{Rel}. (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}) \)
\( \land (\text{TRel} = \{(T1, T2). (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\}) \)
\( \land (\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}) \land \text{preorder } \text{Rel} \)
\( \land \text{weak-reduction-bisimulation } \text{Rel} (\text{STCal Source Target}) \)

by blast

hence operational-corresponding \( \text{TRel} \land \text{preorder } \text{TRel} \land \text{weak-reduction-bisimulation } \text{TRel Target} \)

using OC-wrt-preorder-iff-weak-reduction-bisimulation[where \( \text{TRel}=\text{TRel} \)]

by simp

moreover have \( \exists \text{Rel}. (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}) \)
\( \land \text{rel-weakly-respects-barbs } \text{Rel} (\text{STCalWB SWB TWB}) \)

apply (rule exI) using C1 C6 by blast

hence enc-weakly-respects-barbs

using enc-weakly-respects-barbs-iff-source-target-rel

by simp

moreover hence enc-weakly-respects-barb-set \{success\}

by simp

moreover have rel-weakly-respects-barbs \( \text{TRel TWB} \)

proof auto

fix \( TP \ TQ \ x \ TP' \)
assume \( (TP, TQ) \in \text{TRel} \)
with C2 have \( (\text{TargetTerm } TP, \text{TargetTerm } TQ) \in \text{Rel} \)
by simp

moreover assume \( TP \rightarrow(\text{Calculus TWB})^\ast TP' \) and \( TP'\downarrow<\text{TWB}>x \)

hence \( \text{TargetTerm } TP\downarrow<\text{STCalWB SWB TWB}>x \) using \( \text{STCalWB-reachesBarbST} \)

by blast

ultimately have \( \text{TargetTerm } TQ\downarrow<\text{STCalWB SWB TWB}>x \)

using C6

by blast

thus \( TQ\downarrow<\text{TWB}>x \)

using \( \text{STCalWB-reachesBarbST} \)

by blast

next

fix \( TP \ TQ \ x \ TQ' \)
assume \( (TP, TQ) \in \text{TRel} \)
with C2 have \( (\text{TargetTerm } TP, \text{TargetTerm } TQ) \in \text{Rel} \)
by simp

moreover assume \( TQ \rightarrow(\text{Calculus TWB})^\ast TQ' \) and \( TQ'\downarrow<\text{TWB}>x \)

hence \( \text{TargetTerm } TQ\downarrow<\text{STCalWB SWB TWB}>x \) using \( \text{STCalWB-reachesBarbST} \)

by blast

ultimately have \( \text{TargetTerm } TP\downarrow<\text{STCalWB SWB TWB}>x \)

using C6

by blast

thus \( TP\downarrow<\text{TWB}>x \)

using \( \text{STCalWB-reachesBarbST} \)

by blast

qed

moreover hence rel-weakly-respects-barb-set \( \text{TRel TWB} \) \{success\}

by blast

ultimately show operational-corresponding \( \text{TRel} \land \text{preorder } \text{TRel} \land \text{weak-reduction-bisimulation } \text{TRel Target} \)
\( \land \text{enc-weakly-respects-barbs } \land \text{rel-weakly-respects-barbs } \text{TRel TWB} \)
\( \land \text{enc-weakly-respects-barb-set } \{\text{success}\} \land \text{rel-weakly-respects-barb-set } \text{TRel TWB} \) \{success\}

by fast

qed

An encoding is success sensitive and weakly operational corresponding w.r.t. a correspondence sim-
ulation \( \text{TRel} \) that respects success iff there exists a correspondence simulation that includes \( \text{TRel} \) and
respects success. The same holds if we consider not only success sensitiveness but barb sensitiveness
in general.

lemma (in encoding-wrt-barbs) WOC-SS-wrt-preorder-iff-source-target-rel:  
  fixes success :: 'bars
  and TRel :: ('procT × 'procT) set
  shows (weakly-operational-corresponding TRel ∧ preorder TRel
    ∧ weak-reduction-correspondence-simulation TRel Target
    ∧ enc-weakly-respects-barb-set {success}
    ∧ rel-weakly-respects-barb-set TRel TWB {success})
    = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
    ∧ TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
    ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel)
    ∧ weak-reduction-correspondence-simulation Rel (STCal Source Target) ∧ preorder Rel
    ∧ rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) {success})

proof (rule iffI, (erule conjE)+)
  assume A1: rel-weakly-preserves-barb-set TRel TWB {success}
  and A2: rel-weakly-reflects-barb-set TRel TWB {success}
  and A3: enc-weakly-preserves-barb-set {success}
  and A4: enc-weakly-reflects-barb-set {success}
  and A5: preorder TRel
  from A5 have A6: TRel⁺ = TRel
    using trancl-id[of TRel]
    unfolding preorder-on-def
    by blast
  from A5 A6 have A7: TRel⁺ = TRel
    using reflcl-trancl[of TRel] trancl-id[of TRel]
    unfolding preorder-on-def refl-on-def
    by auto
  define Rel where Rel = indRelRTPO TRel
  hence B1: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
    by (simp add: indRelRTPO.encR)
  from Rel-def A6 have B2: TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
    using indRelRTPO-to-TRel(4)[where TRel=TRel]
    by (auto simp add: indRelRTPO.target)
  from Rel-def A7 have B3: ∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel
    using indRelRTPO-to-TRel(2)[where TRel=TRel]
    trans-closure-of-TRel-refl-cond[where TRel=TRel]
    by simp
  assume operational-complete TRel and weakly-operational-sound TRel
  and weak-reduction-simulation TRel Target
  and ∀ P Q Q'. (P, Q) ∈ TRel ∧ Q' → Target* Q'
    → (∃ P' Q". P → Target* P" ∧ Q' → Target* Q" ∧ (P'', Q'') ∈ TRel)
  with Rel-def A6 A7 have B4: weak-reduction-correspondence-simulation Rel (STCal Source Target)
    using WOC-iff-indRelRTPO-is-reduction-correspondence-simulation[where TRel=TRel]
    by simp
  from Rel-def A5 have B5: preorder Rel
    using indRelRTPO-is-preorder[where TRel=TRel]
    unfolding preorder-on-def
    by blast
  from Rel-def A1 A2 A3 A4 have B6: rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) {success}
    using enc-and-TRel-impl-indRelRTPO-weakly-respects-success[where TRel=TRel]
    and success=success
    by blast
  show (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
    ∧ TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
    ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel)
    ∧ weak-reduction-correspondence-simulation Rel (STCal Source Target) ∧ preorder Rel
    ∧ rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) {success}
    apply (rule ex1I) using B1 B2 B3 B4 B5 B6 by blast
next
  assume ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
ultimately show \( \text{weakly-operational-corresponding} \ TRel \land \text{preorder} \ TRel \)
\land \text{weak-reduction-correspondence-simulation} \ TRel \ Target
\land \text{enc-weakly-respects-barb-set} \ \{\text{success}\} \land \text{rel-weakly-respects-barb-set} \ TRel \ TWB \ \{\text{success}\}
by fast

\[\begin{align*}
\land \ TRel = \{(T1, T2). (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel}\} \\
\land (\forall S T. (\text{SourceTerm} S, \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}) \\
\land \text{weak-reduction-correspondence-simulation} \ Rel \ (\text{STCal Source Target}) \land \text{preorder} \ Rel \\
\land \text{rel-weakly-respects-barb-set} \ Rel \ (\text{STCalWB} \ \text{SWB} \ TWB) \ \{\text{success}\}
\end{align*}\]

from this obtain \( \text{Rel} \) where \( C1: (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}) \)
and \( C2: TRel = \{(T1, T2). (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel}\} \)
and \( C3: (\forall S T. (\text{SourceTerm} S, \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}) \)
and \( C4: \text{weak-reduction-correspondence-simulation} \ Rel \ (\text{STCal Source Target}) \)
and \( C5: \text{preorder} \ Rel \) and \( C6: \text{rel-weakly-respects-barb-set} \ Rel \ (\text{STCalWB} \ \text{SWB} \ TWB) \ \{\text{success}\}
by auto

hence \text{weakly-operational-corresponding} \ TRel \land \text{preorder} \ TRel
\land \text{weak-reduction-correspondence-simulation} \ TRel \ Target
using \text{WOC-wrt-preorder-iff-reduction-correspondence-simulation}[\text{where} TRel=TRel]
by simp

moreover have \( \exists \text{Rel}. (\forall S. (\text{SourceTerm} S, \text{TargetTerm} ([S])) \in \text{Rel}) \)
\land (\text{TRel} = \{(T1, T2). (\text{TargetTerm} T1, \text{TargetTerm} T2) \in \text{Rel}\})
\land (\forall S T. (\text{SourceTerm} S, \text{TargetTerm} T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}) \land \text{preorder} \ Rel
\land \text{weak-reduction-correspondence-simulation} \ Rel \ (\text{STCal Source Target})
by blast

hence \text{enc-weakly-respects-barb-set} \ \{\text{success}\}
using \text{success-sensitive-iff-source-target-rel-weakly-respects-success}
by simp

moreover have \text{rel-weakly-respects-barb-set} \ TRel \ TWB \ \{\text{success}\}

proof auto
fix \( TP TQ TP' \)
assume \((TP, TQ) \in \text{TRel})
with \( C2 \) have \((\text{TargetTerm} TP, \text{TargetTerm} TQ) \in \text{Rel})
by simp
moreover assume \( TP \rightarrow (\text{Calculus} \ TWB) * TP' \) and \( TP' < TWB > \text{success} \)
hence \( \text{TargetTerm} \ TP \downarrow < \text{STCalWB} \ SWB \ TWB > \text{success} \)
using \text{STCalWB-reachesBarbST}
by blast
ultimately have \( \text{TargetTerm} \ TQ \downarrow < \text{STCalWB} \ SWB \ TWB > \text{success} \)
using \( C6 \)
by blast
thus \( TQ < TWB > \text{success} \)
using \text{STCalWB-reachesBarbST}
by blast

next
fix \( TP TQ TP' \)
assume \((TP, TQ) \in \text{TRel})
with \( C2 \) have \((\text{TargetTerm} TP, \text{TargetTerm} TQ) \in \text{Rel})
by simp
moreover assume \( TQ \rightarrow (\text{Calculus} \ TWB) * TQ' \) and \( TQ' < TWB > \text{success} \)
hence \( \text{TargetTerm} TQ \downarrow < \text{STCalWB} \ SWB \ TWB > \text{success} \)
using \text{STCalWB-reachesBarbST}
by blast
ultimately have \( \text{TargetTerm} \ TP \downarrow < \text{STCalWB} \ SWB \ TWB > \text{success} \)
using \( C6 \)
by blast
thus \( TP < TWB > \text{success} \)
using \text{STCalWB-reachesBarbST}
by blast

qed
ultimately show \text{weakly-operational-corresponding} \ TRel \land \text{preorder} \ TRel
\land \text{weak-reduction-correspondence-simulation} \ TRel \ Target
\land \text{enc-weakly-respects-barb-set} \ \{\text{success}\} \land \text{rel-weakly-respects-barb-set} \ TRel \ TWB \ \{\text{success}\}
by fast

255
proof

 fixes success :: "barbs"

 and TRel :: ('procT × 'procT) set

 shows (weakly-operational-corresponding TRel ∧ preorder TRel

 ∧ weak-reduction-correspondence-simulation TRel Target

 ∧ enc-weakly-respects-barbs ∧ enc-weakly-respects-barb-set {success}

 ∧ rel-weakly-respects-barbs TRel TWB ∧ rel-weakly-respects-barb-set TRel TWB {success})

 = (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)

 ∧ TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}

 ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → (|[S]|, T) ∈ TRel)

 ∧ weak-reduction-correspondence-simulation Rel (STCal Source Target) ∧ preorder Rel

 ∧ rel-weakly-respects-barbs Rel (STCalWB SWB TWB)

 ∧ rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) {success})

proof (rule iffI, (erule conjE)+)

 assume A1: rel-weakly-preserves-barb-set TRel TWB {success}

 and A2: rel-weakly-reflects-barb-set TRel TWB {success}

 and A3: enc-weakly-preserves-barb-set {success}

 and A4: enc-weakly-reflects-barb-set {success}

 and A5: preorder TRel

 and A1': rel-weakly-preserves-barbs TRel TWB and A2': rel-weakly-reflects-barbs TRel TWB

 and A3': enc-weakly-preserves-barbs and A4': enc-weakly-reflects-barbs

 from A5 have A6: TRel⁺ = TRel

 using preorder-on-def

 unfolding preorder-on-def

 by blast

 from A5 A6 have A7: TRel⁺ = TRel

 using refl-trancl[of TRel] trancl-id[of TRel]

 unfolding preorder-on-def refl-on-def

 by auto

 define Rel where Rel = indRelRTPO TRel

 hence B1: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel

 by (simp add: indRelRTPO.encR)

 from Rel-def A6 have B2: TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}

 using indRelRTPO-to-TRel(4)[where TRel= TRel]

 by (auto simp add: indRelRTPO.target)

 from Rel-def A7 have B3: ∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel

 using indRelRTPO-to-TRel(2)[where TRel= TRel]

 trans-closure-of-TRel-refl-conv[where TRel= TRel]

 by simp

 assume operational-complete TRel and weakly-operational-sound TRel

 and weak-reduction-simulation TRel Target

 and ∀ P Q Q' (P, Q) ∈ TRel ∧ Q → Target* Q'

 → (∃ P'' Q''. P → Targets* P'' ∧ Q'' → Target* Q'' ∧ (P'', Q'') ∈ TRel)

 with Rel-def A6 A7 have B4: weak-reduction-correspondence-simulation Rel (STCal Source Target)

 using WOC-iff-indRelRTPO-is-reduction-correspondence-simulation[where TRel= TRel]

 by simp

 from Rel-def A5 have B5: preorder Rel

 using indRelRTPO-is-preorder[where TRel= TRel]

 unfolding preorder-on-def

 by blast

 from Rel-def A1 A2 A3 A4 have B6: rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) {success}

 using enc-and-TRel-impl-indRelRTPO-weakly-respects-success[where TRel= TRel]

 and success=success

 by blast

 from Rel-def A1' A2' A3' A4' have B7: rel-weakly-respects-barbs Rel (STCalWB SWB TWB)

 using enc-and-TRel-impl-indRelRTPO-weakly-respects-barbs[where TRel= TRel]

 by blast

 show ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)

 ∧ TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
∀ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel)
∧ weak-reduction-correspondence-simulation Rel (STCal Source Target) ∧ preorder Rel
∧ rel-weakly-respects-barbs Rel (STCalWB SWB TWB)
∧ rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) {success}
apply (rule exf) using B1 B2 B4 B5 B6 B7 by blast
next
assume ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
∧ TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel)
∧ weak-reduction-correspondence-simulation Rel (STCal Source Target) ∧ preorder Rel
∧ rel-weakly-respects-barbs Rel (STCalWB SWB TWB)
∧ rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) {success}
from this obtain Rel where C1: (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
and C2: TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
and C3: (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel)
and C4: weak-reduction-correspondence-simulation Rel (STCal Source Target)
and C5: preorder Rel and C7: rel-weakly-respects-barbs Rel (STCalWB SWB TWB)
by auto
from C1 C2 C3 C4 C5 have ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
∧ (TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel})
∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel) ∧ preorder Rel
∧ weak-reduction-correspondence-simulation Rel (STCal Source Target)
by blast
hence weakly-operational-corresponding TRel ∧ preorder TRel
∧ weak-reduction-correspondence-simulation TRel Target
using WOC-wrt-preorder-iff-reduction-correspondence-simulation[where TRel=TRel]
by simp
moreover have ∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
∧ rel-weakly-respects-barbs Rel (STCalWB SWB TWB)
apply (rule exf) using C1 C7 by blast
hence D1: enc-weakly-respects-barbs
using enc-weakly-respects-barbs-iff-source-target-rel
by simp
moreover from D1 have enc-weakly-respects-barb-set {success}
by simp
moreover have D2: rel-weakly-respects-barbs TRel TWB
proof auto
fix TP TQ x TP'
assume (TP, TQ) ∈ TRel
with C2 have (TargetTerm TP, TargetTerm TQ) ∈ Rel
by simp
moreover assume TP →→ (Calculus TWB)* TP' and TP'↓<TWB>x
hence TargetTerm TP↓<STCalWB SWB TWB>x
using STCalWB-reachesBarbST
by blast
ultimately have TargetTerm TQ↓<STCalWB SWB TWB>x
using C7
by blast
thus TQ↓<TWB>x
using STCalWB-reachesBarbST
by blast
next
fix TP TQ x TQ'
assume (TP, TQ) ∈ TRel
with C2 have (TargetTerm TP, TargetTerm TQ) ∈ Rel
by simp
moreover assume TQ →→ (Calculus TWB)* TQ' and TQ'↓<TWB>x
hence TargetTerm TQ↓<STCalWB SWB TWB>x
using STCalWB-reachesBarbST
by blast
ultimately have TargetTerm TP↓<STCalWB SWB TWB>x
An encoding is strongly success sensitive and strongly operational corresponding w.r.t. a strong bisimulation TRel that strongly respects success if there exists a strong bisimulation that includes TRel and strongly respects success. The same holds if we consider not only strong success sensitiveness but strong barb sensitiveness in general.

**Lemma (in encoding-wrt-barbs)** SOC-SS-wrt-preorder-iff-source-target-rel:

- fixes `success` :: `'barbs`
- and `TRel` := `(procT × procT)` set
- shows (strongly-operational-corresponding TRel ∧ preorder TRel
  ∧ strong-reduction-bisimulation TRel Target
  ∧ enc-respects-barb-set {success} ∧ rel-repects-barb-set TRel TWB {success})

- proof (rule `iffI`, (erule conjE)+)
  - assume `A1`: rel-preserves-barb-set TRel TWB {success}
    - and `A2`: rel-reflects-barb-set TRel TWB {success}
    - and `A3`: enc-preserves-barb-set {success} and `A4`: enc-reflects-barb-set {success}
    - and `A5`: preorder TRel
  - from `A5` have `A6`: `TRel` = TRel
    - using trancl-id[of TRel]
    - unfolding preorder-on-def
    - by blast
  - from `A5 A6` have `A7`: `TRel` = TRel
    - using reflcl-trancl[of TRel] trancl-id[of TRel]
    - unfolding preorder-on-def refl-on-def
    - by auto
  - define `Rel` where `Rel` = `indRelRTPO TRel`
  - hence `B1`: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel
    - by (simp add: `indRelRTPO.encR`)
  - from `Rel-def A6` have `B2`: `TRel` = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
    - using `indRelRTPO-to-TRel[4]`[where `TRel`=`TRel`]
    - by (auto simp add: `indRelRTPO.target`)
  - from `Rel-def A7` have `B3`: ∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel
    - using `indRelRTPO-to-TRel[2]`[where `TRel`=`TRel`]
    - trans-closure-of-TRel-refl-cond[where `TRel`=`TRel`]
    - by simp
  - assume strongly-operational-complete TRel and strongly-operational-sound TRel
    - and strong-reduction-simulation TRel Target
    - and ∀ P Q Q'. (P, Q) ∈ TRel ∧ Q → Target Q' → (∃ P'. P → Target P' ∧ (P', Q') ∈ TRel)
  - with `Rel-def A6 A7` have `B4`: strong-reduction-bisimulation Rel (STCal Source Target)
    - using SOC-iff-indRelRTPO-is-strong-reduction-bisimulation[where `TRel`=`TRel`]
    - by simp
  - from `Rel-def A5` have `B5`: preorder Rel
using \textit{indRelRTPO-is-preorder}[\textbf{where} \ TRel = TRel]

\textbf{unfolding} \ \textit{preorder-on-def}

by \textbf{blast}

from \textit{Rel-def A1 A2 A3 A4} \textbf{have} \textit{B6: rel-respects-barb-set Rel (STCalWB SWB TWB) \{success\}}

using \textit{enc-and-TRel-impl-indRelRTPO-respects-success}[\textbf{where} \ TRel = TRel \ \textbf{and} \ \textit{success} = \textit{success}]

by \textbf{blast}

\textbf{show} \ \exists \textit{Rel}. \ (\forall \ S. \ (\textit{SourceTerm S}, \ \textit{TargetTerm ([S])}) \in \textit{Rel})

\land \ TRel = \{\{T1, T2\}. \ (\textit{TargetTerm T1}, \ \textit{TargetTerm T2}) \in \textit{Rel}\}

\land \ (\forall \ S \ T. \ (\textit{SourceTerm S}, \ \textit{TargetTerm T}) \in \textit{Rel} \rightarrow ([S], T) \in \textit{Treld})

\land \ \textit{strong-reduction-bisimulation \ Rel (STCal Source Target)} \ \land \ \textit{preorder Rel}

\land \ \textit{rel-respects-barb-set Rel (STCalWB SWB TWB) \{success\}}

\textbf{apply} (\textbf{rule exI}) \ \textbf{using} \ B1 B2 B3 B4 B5 B6 \ \textbf{by} \ \textbf{blast}

\textbf{next}

\textbf{assume} \ \exists \textit{Rel}. \ (\forall \ S. \ (\textit{SourceTerm S}, \ \textit{TargetTerm ([S])}) \in \textit{Rel})

\land \ TRel = \{\{T1, T2\}. \ (\textit{TargetTerm T1}, \ \textit{TargetTerm T2}) \in \textit{Rel}\}

\land \ (\forall \ S \ T. \ (\textit{SourceTerm S}, \ \textit{TargetTerm T}) \in \textit{Rel} \rightarrow ([S], T) \in \textit{Treld})

\land \ \textit{strong-reduction-bisimulation \ Rel (STCal Source Target)} \ \land \ \textit{preorder Rel}

\land \ \textit{rel-respects-barb-set Rel (STCalWB SWB TWB) \{success\}}

\textbf{from this obtain} \ \textbf{Rel} \ \textit{where} \ C1: \ (\forall \ S. \ (\textit{SourceTerm S}, \ \textit{TargetTerm ([S])}) \in \textit{Rel})

and \ C2: \ TRel = \{\{T1, T2\}. \ (\textit{TargetTerm T1}, \ \textit{TargetTerm T2}) \in \textit{Rel}\}

and \ C3: \ (\forall \ S \ T. \ (\textit{SourceTerm S}, \ \textit{TargetTerm T}) \in \textit{Rel} \rightarrow ([S], T) \in \textit{Treld})

and \ C4: \ \textit{strong-reduction-bisimulation \ Rel (STCal Source Target)} \ \textbf{and} \ C5: \ \textit{preorder Rel}

and \ C6: \ \textit{rel-respects-barb-set Rel (STCalWB SWB TWB) \{success\}}

by \textbf{auto}

from \ C1 \ C2 \ C3 \ C4 \ C5 \ \textbf{have} \ \exists \textit{Rel}. \ (\forall \ S. \ (\textit{SourceTerm S}, \ \textit{TargetTerm ([S])}) \in \textit{Rel})

\land \ (\textit{Treld} = \{\{T1, T2\}. \ (\textit{TargetTerm T1}, \ \textit{TargetTerm T2}) \in \textit{Rel}\})

\land \ (\forall \ S \ T. \ (\textit{SourceTerm S}, \ \textit{TargetTerm T}) \in \textit{Rel} \rightarrow ([S], T) \in \textit{Treld}) \ \land \ \textit{preorder Rel}

\land \ \textit{strong-reduction-bisimulation \ Rel (STCal Source Target)}

by \textbf{blast}

\textbf{hence} \ \textit{strongly-operational-corresponding \ TRel} \ \land \ \textit{preorder TRel}

\land \ \textit{strong-reduction-bisimulation \ TRel Target}

\textbf{using} \ \textit{SOC-wrt-preorder-iff-strong-reduction-bisimulation}[\textbf{where} \ TRel = TRel]

by \textbf{simp}

\textbf{moreover have} \ \exists \textit{Rel}. \ (\forall \ S. \ (\textit{SourceTerm S}, \ \textit{TargetTerm ([S])}) \in \textit{Rel})

\land \ (\textit{Treld} = \{\{T1, T2\}. \ (\textit{TargetTerm T1}, \ \textit{TargetTerm T2}) \in \textit{Rel}\})

\land \ (\forall \ S \ T. \ (\textit{SourceTerm S}, \ \textit{TargetTerm T}) \in \textit{Rel} \rightarrow ([S], T) \in \textit{Treld}) \ \land \ \textit{preorder Rel}

\land \ \textit{strong-reduction-bisimulation \ Rel (STCal Source Target)}

by \textbf{simp}

\textbf{hence} \ \textit{enc-respects-barb-set \ \{success\}}

\textbf{using} \ \textit{success-sensitive-iff-source-target-rel-respects-success}

by \textbf{simp}

\textbf{moreover have} \ \textit{rel-respects-barb-set \ TRel TWB \ \{success\}}

\textbf{proof}

\textbf{auto}

\textbf{fix} \ TP \ TQ

\textbf{assume} \ (TP, TQ) \in \textit{TRel}

\textbf{with} \ C2 \ \textbf{have} \ (\textit{TargetTerm TP}, \ \textit{TargetTerm TQ}) \in \textit{Rel}

by \textbf{simp}

\textbf{moreover assume} \ TP \downarrow <TWB> \success

\textbf{hence} \ \textit{TargetTerm TP} \downarrow <STCalWB SWB TWB> \success

\textbf{using} \ \textit{STCalWB-hasBarbST}

by \textbf{blast}

\textbf{ultimately have} \ \textit{TargetTerm TQ} \downarrow <STCalWB SWB TWB> \success

\textbf{using} \ C6

by \textbf{blast}

\textbf{thus} \ TQ \downarrow <TWB> \success

\textbf{using} \ \textit{STCalWB-hasBarbST}

by \textbf{blast}

\textbf{next}

\textbf{fix} \ TP \ TQ

\textbf{assume} \ (TP, TQ) \in \textit{TRel}

\textbf{with} \ C2 \ \textbf{have} \ (\textit{TargetTerm TP}, \ \textit{TargetTerm TQ}) \in \textit{Rel}

by \textbf{simp}

\textbf{moreover assume} \ TQ \downarrow <TWB> \success
proof

lemma (in encoding-wrt-barbs) SOC-SS-RB-wrt-preorder-iff-source-target-rel:

fixes success :: 'barbs

and TRel ≝ ('procT × 'procT) set

shows (strongly-operational-corresponding TRel ∧ preorder TRel
∧ strong-reduction-bisimulation TRel Target
∧ enc-respects-barb-set {success} ∧ rel-respects-barb-set TRel TWB {success})

= (∃ Rel. (∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel)
∧ TRel = {(T1, T2), (TargetTerm T1, TargetTerm T2) ∈ Rel}
∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel)
∧ strong-reduction-bisimulation Rel (STCal Source Target) ∧ preorder Rel
∧ rel-respects-barbs Rel (STCalWB SWB TWB)
∧ rel-respects-barb-set Rel (STCalWB SWB TWB) {success})

proof (rule iffI, (erule conjE)+)

assume A1: rel-preserves-barbs TRel TWB

and A2: rel-reflects-barbs TRel TWB

and A3: enc-preserves-barbs TRel TWB

and A4: enc-reflects-barbs TRel TWB

and A5: preorder TRel

from A5 have A6: TRel⁺ = TRel

using rel-preserves-barbs

unfolding preorder-on-def

by blast

from A5 have A7: TRel⁺ = TRel

using rel-reflects-barbs

unfolding preorder-on-def refl-on-def

by auto

define Rel where Rel = indRelRTPO TRel

hence B1: ∀ S. (SourceTerm S, TargetTerm ([S])) ∈ Rel

by (simp add: indRelRTPO.encR)

from Rel-def A6 have B2: TRel = {(T1, T2), (TargetTerm T1, TargetTerm T2) ∈ Rel}

using indRelRTPO-to-TRel(4)[where TRel=TRel]

by (auto simp add: indRelRTPO.target)

from Rel-def A7 have B3: ∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel

using indRelRTPO-to-TRel(2)[where TRel=TRel]

trans-closure-of-TRel-refl-cond[where TRel=TRel]

by simp

assume strongly-operational-complete TRel and strongly-operational-sound TRel

and preorder-simulation TRel Target

and ∀ P Q Q′. (P, Q) ∈ TRel ∧ Q −→ Target Q′ → (∃ P′. P −→ Target P′ ∧ (P′, Q′) ∈ TRel)

with Rel-def A6 A7 have B4: strong-reduction-bisimulation Rel (STCal Source Target)

using SOC-iff-indRelRTPO-is-strong-reduction-bisimulation[where TRel=TRel]

by simp

from Rel-def A5 have B5: preorder Rel

using indRelRTPO-is-preorder[where TRel=TRel]

unfolding preorder-on-def

by blast
from Rel-def A1 A2 A3 A4 have B6: rel-respects-barbs Rel (STCalWB SWB TWB)
using enc-and-TRel-impl-indRelRTPO-respects-barbs|where TRel = TRel
by blast
hence B7: rel-respects-barb-set Rel (STCalWB SWB TWB) \{success\}
by blast
show \exists Rel. (\forall S. (SourceTerm S, TargetTerm ([S])) \in Rel)
\land TRel = \{(T1, T2), (TargetTerm T1, TargetTerm T2) \in Rel\}
\land (\forall S T. (SourceTerm S, TargetTerm T) \in Rel \rightarrow ([S], T) \in TRel)
\land strongly-reduction-bisimulation Rel (STCal Source Target) \land preorder Rel
\land rel-respects-barbs Rel (STCalWB SWB TWB)
\land rel-respects-barb-set Rel (STCalWB SWB TWB) \{success\}
apply (rule exI) using B1 B2 B3 B4 B5 B6 by blast
next
assume \exists Rel. (\forall S. (SourceTerm S, TargetTerm ([S])) \in Rel)
\land TRel = \{(T1, T2), (TargetTerm T1, TargetTerm T2) \in Rel\}
\land (\forall S T. (SourceTerm S, TargetTerm T) \in Rel \rightarrow ([S], T) \in TRel)
\land strongly-reduction-bisimulation Rel (STCal Source Target) \land preorder Rel
\land rel-respects-barbs Rel (STCalWB SWB TWB)
\land rel-respects-barb-set Rel (STCalWB SWB TWB) \{success\}
from this obtain Rel where C1: (\forall S. (SourceTerm S, TargetTerm ([S])) \in Rel)
and C2: TRel = \{(T1, T2), (TargetTerm T1, TargetTerm T2) \in Rel\}
and C3: (\forall S T. (SourceTerm S, TargetTerm T) \in Rel \rightarrow ([S], T) \in TRel)
and C4: strongly-reduction-bisimulation Rel (STCal Source Target) and C5: preorder Rel
and C6: rel-respects-barbs Rel (STCalWB SWB TWB)
by auto
from C1 C2 C3 C4 C5 have \exists Rel.(\forall S. (SourceTerm S, TargetTerm ([S])) \in Rel)
\land (TRel = \{(T1, T2), (TargetTerm T1, TargetTerm T2) \in Rel\})
\land (\forall S T. (SourceTerm S, TargetTerm T) \in Rel \rightarrow ([S], T) \in TRel)
\land strongly-reduction-bisimulation Rel (STCal Source Target)
by blast
hence strongly-operational-corresponding TRel \land preorder TRel
\land strongly-reduction-bisimulation TRel Target
using SOC-wrt-preorder-iff-strong-reduction-bisimulation|where TRel = TRel
by simp
moreover have \exists Rel. (\forall S. (SourceTerm S, TargetTerm ([S])) \in Rel)
\land rel-respects-barbs Rel (STCalWB SWB TWB)
apply (rule exI) using C1 C6 by blast
hence enc-respects-barbs
using enc-respects-barbs-iff-source-target-rel
by simp
moreover hence enc-respects-barb-set \{success\}
by simp
moreover have rel-respects-barbs TRel TWB
proof auto
fix TP TQ x
assume (TP, TQ) \in TRel
with C2 have (TargetTerm TP, TargetTerm TQ) \in Rel
by simp
moreover assume TP \downarrow TWB \circ x
hence TargetTerm TP \downarrow STCalWB SWB TWB \circ x
using STCalWB-hasBarbST
by blast
ultimately have TargetTerm TQ \downarrow STCalWB SWB TWB \circ x
using C6
by blast
thus TQ \downarrow TWB \circ x
using STCalWB-hasBarbST
by blast
next
fix TP TQ x
assume (TP, TQ) \in TRel
with $C_2$ have $(\text{TargetTerm } TP, \text{TargetTerm } TQ) \in \text{Rel}$
by simp
moreover assume $TQ \downarrow <\text{STCalWB } SWB \text{ TWB}>$
hence $\text{TargetTerm } TQ \downarrow <\text{STCalWB } SWB \text{ TWB}>$
using $\text{STCalWB-hasBarbST}$
by blast
ultimately have $\text{TargetTerm } TP \downarrow <\text{STCalWB } SWB \text{ TWB}>$
using $C_6$
by blast
thus $TP \downarrow <\text{TWB}>$
using $\text{STCalWB-hasBarbST}$
by blast
qed
moreover hence $\text{rel-respects-barb-set } \text{TRel } \text{TWB } \{\text{success}\}$
by blast
ultimately show strongly-operational-corresponding $\text{TRel } \land \text{preorder } \text{TRel}$
$\land \text{enc-respects-barbs } \land \text{rel-respects-barbs } \text{TRel } \text{TWB}$
$\land \text{enc-respects-barb-set } \{\text{success}\} \land \text{rel-respects-barb-set } \text{TRel } \text{TWB } \{\text{success}\}$
by fast
qed
Next we also add divergence reflection to operational correspondence and success sensitiveness.

**Lemma (in encoding)** $\text{enc-and-TRelimpl-indRelRTPO-reflect-divergence:}$
fixes $\text{TRel} : (\text{procT } \times \text{procT}) \text{ set}$
assumes $\text{encRD}: \text{enc-reflects-divergence}$
and $\text{trelRD}: \text{rel-reflects-divergence } \text{TRel } \text{Target}$
shows $\text{rel-reflects-divergence } (\text{indRelRTPO } \text{TRel}) (\text{STCal Source Target})$
proof auto
fix $P$ $Q$
assume $P \triangleright\triangleright_{\text{RT}<\text{TRel}>} Q$ and $Q \longmapsto\omega (\text{STCal Source Target})$
thus $P \longmapsto\omega (\text{STCal Source Target})$
proof induct
  case (\text{encR } S)
  assume $\text{TargetTerm } [\{S\}] \longmapsto\omega (\text{STCal Source Target})$
hence $[S] \longmapsto\omega (\text{Target})$
  by (simp add: \text{STCal-divergent(2)})
  with $\text{encRD have } S \longmapsto\omega (\text{Source})$
  by simp
  thus $\text{SourceTerm } S \longmapsto\omega (\text{STCal Source Target})$
  by (simp add: \text{STCal-divergent(1)})
next
  case (\text{source } S)
  assume $\text{SourceTerm } S \longmapsto\omega (\text{STCal Source Target})$
  thus $\text{SourceTerm } S \longmapsto\omega (\text{STCal Source Target})$
  by simp
next
  case (\text{target } T1 T2)
  assume $(T1, T2) \in \text{TRel}$
  moreover assume $\text{TargetTerm } T2 \longmapsto\omega (\text{STCal Source Target})$
hence $T2 \longmapsto\omega (\text{Target})$
  by (simp add: \text{STCal-divergent(2)})
  ultimately have $T1 \longmapsto\omega (\text{Target})$
  using $\text{trelRD}$
  by blast
  thus $\text{TargetTerm } T1 \longmapsto\omega (\text{STCal Source Target})$
  by (simp add: \text{STCal-divergent(2)})
next
  case (\text{trans } P Q R)
  assume $R \longmapsto\omega (\text{STCal Source Target})$
  and $R \longmapsto\omega (\text{STCal Source Target}) \Rightarrow Q \longmapsto\omega (\text{STCal Source Target})$

262
and $Q \longrightarrow (\text{STCal Source Target})\omega \Rightarrow P \longrightarrow (\text{STCal Source Target})\omega$

thus $P \longrightarrow (\text{STCal Source Target})\omega$

by simp

qed

qed

lemma (in encoding-wrt-barbs) OC-SS-DR-iff-source-target-rel:

fixes success :: 'a barbs

and $T_{\text{Rel}}$ :: ('procT × 'procT) set

shows (operational-corresponding ($T_{\text{Rel}}^*$))

\begin{itemize}
  \item weak-reduction-bisimulation ($T_{\text{Rel}}^+$) Target
  \item enc-weakly-respects-barb-set {success}
  \item rel-weakly-respects-barb-set $T_{\text{Rel}}$ TWB {success}
  \item enc-reflects-divergence ∧ rel-reflects-divergence $T_{\text{Rel}}$ Target)
\end{itemize}

= ($\exists$ S. (SourceTerm S, TargetTerm ([S])) ∈ $\text{Rel}$)

∧ ($\forall$ T1 T2. (T1, T2) ∈ $T_{\text{Rel}}$ → (TargetTerm T1, TargetTerm T2) ∈ Rel)

∧ ($\forall$ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ $T_{\text{Rel}}$ → (T1, T2) ∈ $T_{\text{Rel}}^+$)

∧ ($\forall$ S T. (SourceTerm S, TargetTerm T) ∈ $T_{\text{Rel}}$ → ([S], T) ∈ $T_{\text{Rel}}^*$)

∧ weak-reduction-bisimulation Rel (STCal Source Target)

∧ rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) {success}

∧ rel-reflects-divergence Rel (STCal Source Target)

proof (rule iffI, (erule conjE)+)

assume $A_1$: rel-weakly-preserves-barb-set $T_{\text{Rel}}$ TWB {success}

and $A_2$: rel-weakly-reflects-barb-set $T_{\text{Rel}}$ TWB {success}

and $A_3$: enc-weakly-preserves-barb-set {success}

and $A_4$: enc-weakly-reflects-barb-set {success}

and $A_5$: rel-reflects-divergence $T_{\text{Rel}}$ Target and $A_6$: enc-reflects-divergence

define Rel where $\text{Rel} = \text{indRelRTPO} T_{\text{Rel}}$

hence $B_1$: $\forall$ S. (SourceTerm S, TargetTerm ([S])) ∈ $\text{Rel}$

by (simp add: indRelRTPO.encR)

from Rel-def have $B_2$: $\forall$ T1 T2. (T1, T2) ∈ $T_{\text{Rel}}$ → (TargetTerm T1, TargetTerm T2) ∈ Rel

by (simp add: indRelRTPO.target)

from Rel-def have $B_3$: $\forall$ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ $T_{\text{Rel}}$ → (T1, T2) ∈ $T_{\text{Rel}}^+$

by (simp add: indRelRTPO-to-TRel(4)[where $T_{\text{Rel}}$=TRel])

from Rel-def have $B_4$: $\forall$ S T. (SourceTerm S, TargetTerm T) ∈ $T_{\text{Rel}}$ → ([S], T) ∈ $T_{\text{Rel}}^*$

using indRelRTPO-to-TRel(2)[where $T_{\text{Rel}}$=TRel]

trans-closure-of-$T_{\text{Rel}}$-refl-cond[where $T_{\text{Rel}}$=TRel]

by simp

assume operational-complete ($T_{\text{Rel}}^*$)

and operational-sound ($T_{\text{Rel}}^*$)

and weak-reduction-simulation ($T_{\text{Rel}}^+$) Target

and $\forall$ P Q Q'. (P, Q) ∈ $T_{\text{Rel}}^+$ ∧ Q → Target* Q'

→ ($\exists$ P'. P → Target* P' ∧ (P', Q') ∈ $T_{\text{Rel}}^+$)

with Rel-def have $B_5$: weak-reduction-bisimulation Rel (STCal Source Target)

using OC-iff-indRelRTPO-is-weak-reduction-bisimulation[where $T_{\text{Rel}}$=TRel]

by simp

from Rel-def $A_1$ $A_2$ $A_3$ $A_4$ have $B_6$: rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) {success}

using enc-and-TRel-impl-indRelRTPO-weakly-respects-success[where $T_{\text{Rel}}$=TRel]

and success=success

by blast

from Rel-def $A_5$ $A_6$ have $B_7$: rel-reflects-divergence Rel (STCal Source Target)

using enc-and-TRel-impl-indRelRTPO-reflect-divergence[where $T_{\text{Rel}}$=TRel]

by blast

show $\exists$ Rel. ($\forall$ S. (SourceTerm S, TargetTerm ([S])) ∈ $\text{Rel}$)

∧ ($\forall$ T1 T2. (T1, T2) ∈ $T_{\text{Rel}}$ → (TargetTerm T1, TargetTerm T2) ∈ Rel)

∧ ($\forall$ T1 T2. (TargetTerm T1, TargetTerm T2) ∈ $T_{\text{Rel}}$ → (T1, T2) ∈ $T_{\text{Rel}}^+$)

∧ ($\forall$ S T. (SourceTerm S, TargetTerm T) ∈ $T_{\text{Rel}}$ → ([S], T) ∈ $T_{\text{Rel}}^*$)

∧ weak-reduction-bisimulation Rel (STCal Source Target)

∧ rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) {success}

∧ rel-reflects-divergence Rel (STCal Source Target)

apply (rule exI) using $B_1$ $B_2$ $B_3$ $B_4$ $B_5$ $B_6$ $B_7$ by blast

263
\[\begin{align*}
\exists \text{Rel}. & \ (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}) \\
& \land (\forall T1, T2. (T1, T2) \in \text{TRel} \rightarrow (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}) \\
& \land (\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}^+) \\
& \land \text{weak-reduction-bisimulation Rel (STCal Source Target)} \\
& \land \text{rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) \{success\}} \\
& \land \text{rel-reflects-divergence Rel (STCal Source Target)}
\end{align*}\]

\[\begin{align*}
\text{from this obtain Rel where } C1: & \ (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}) \\
& \land C2: (\forall T1, T2. (T1, T2) \in \text{TRel} \rightarrow (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}) \\
& \land C3: (\forall T1, T2. (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel} \rightarrow (T1, T2) \in \text{TRel}^+) \\
& \land C4: (\forall S T. (\text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}^+) \\
& \land C5: \text{weak-reduction-bisimulation Rel (STCal Source Target)} \\
& \land C6: \text{rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) \{success\}} \\
& \land C7: \text{rel-reflects-divergence Rel (STCal Source Target)}
\end{align*}\]

\[\begin{align*}
\text{by auto}
\end{align*}\]

\[\begin{align*}
\text{hence operational-corresponding (TRel\(^+\))} & \land \text{weak-reduction-bisimulation (TRel\(^+\)) Target} \\
& \text{using OC-iff-weak-reduction-bisimulation[where TRel=TRel]} \\
& \text{by auto}
\end{align*}\]

\[\begin{align*}
\text{moreover have } \exists \text{Rel}. (\forall S. (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}) \\
& \land \text{rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) \{success\}} \\
& \land \text{rel-reflects-divergence Rel (STCal Source Target)}
\end{align*}\]

\[\begin{align*}
\text{apply (rule exI) using C1 C6 C7 by blast}
\end{align*}\]

\[\begin{align*}
\text{hence enc-weakly-respects-barb-set \{success\} \land enc-reflects-divergence} \\
& \text{using WSS-DR-iff-source-target-rel} \\
& \text{by auto}
\end{align*}\]

\[\begin{align*}
\text{moreover have rel-weakly-respects-barb-set TRel TWB \{success\}}
\end{align*}\]

\[\begin{align*}
\text{proof auto}
\end{align*}\]

\[\begin{align*}
\text{fix TP TQ TP' with C2 have (TargetTerm TP, TargetTerm TQ) \in \text{Rel}} \\
& \text{by simp}
\end{align*}\]

\[\begin{align*}
\text{moreover assume TP \rightarrow (Calculus TWB)* TP' and TP'↓<TWB>success} \\
& \text{hence TargetTerm TP\downarrow<STCalWB SWB TWB>success} \\
& \text{using STCalWB-reachingBarbST} \\
& \text{by blast}
\end{align*}\]

\[\begin{align*}
\text{ultimately have TargetTerm TQ\downarrow<STCalWB SWB TWB>success} \\
& \text{using C6 by blast} \\
& \text{thus TQ\downarrow<TWB>success} \\
& \text{using STCalWB-reachingBarbST by blast}
\end{align*}\]

\[\begin{align*}
\text{next}
\end{align*}\]

\[\begin{align*}
\text{fix TP TQ TQ' with C2 have (TargetTerm TP, TargetTerm TQ) \in \text{Rel}} \\
& \text{by simp}
\end{align*}\]

\[\begin{align*}
\text{moreover assume TQ \rightarrow (Calculus TWB)* TQ' and TQ'↓<TWB>success} \\
& \text{hence TargetTerm TQ\downarrow<STCalWB SWB TWB>success} \\
& \text{using STCalWB-reachingBarbST by blast}
\end{align*}\]

\[\begin{align*}
\text{ultimately have TargetTerm TP\downarrow<STCalWB SWB TWB>success} \\
& \text{using C6 by blast}
\end{align*}\]
thus $TP\downarrow <TWB>success$

using $STCalWB$-reachesBarbST

by blast

qed

moreover from C2 C7 have rel-reflects-divergence $T\text{Rel}$ Target

using $STCal$-divergent(2)

by blast

ultimately show operational-corresponding ($T\text{Rel}^+$)

\& weak-reduction-bisimulation ($T\text{Rel}^+$) Target

\& enc-weakly-respects-barb-set {success} \& rel-weakly-respects-barb-set $T\text{Rel}$ $TWB$ {success}

\& enc-reflects-divergence \& rel-reflects-divergence $T\text{Rel}$ Target

by fast

qed

\textbf{lemma (in encoding-wrt-barbs) WOC-SS-DR-wrt-preorder-iff-source-target-rel:}

\textbf{fixes success :: 'barbs}

\textbf{and} $T\text{Rel} :: (\text{proc} T \times \text{proc} T) \text{ set}$

\textbf{shows} (weakly-operational-corresponding $T\text{Rel}$ \& preorder $T\text{Rel}$

\& weak-reduction-correspondence-simulation $T\text{Rel}$ Target

\& enc-weakly-respects-barb-set {success}

\& rel-weakly-respects-barb-set $T\text{Rel}$ $TWB$ {success}

\& enc-reflects-divergence \& rel-reflects-divergence $T\text{Rel}$ Target)

= (\exists $Rel$. (\forall S. (SourceTerm S, TargetTerm ([S])) \in Rel)

\& $T\text{Rel} = \{(T1, T2), (TargetTerm T1, TargetTerm T2) \in Rel\}$

\& (\forall S T. (SourceTerm S, TargetTerm T) \in Rel \rightarrow ([S], T) \in TRel)

\& weak-reduction-correspondence-simulation $Rel$ ($STCal$ Source Target) \& preorder $Rel$

\& rel-weakly-respects-barb-set $Rel$ ($STCalWB$ $SWB$ $TWB$) {success}

\& rel-reflects-divergence $Rel$ ($STCal$ Source Target))

\textbf{proof (rule iffI, (rule conjE)+)}

\textbf{assume A1: rel-weakly-preserves-barb-set $T\text{Rel}$ $TWB$ {success}}

\textbf{and A2: rel-weakly-reflects-barb-set $T\text{Rel}$ $TWB$ {success}}

\textbf{and A3: enc-weakly-preserves-barb-set {success}}

\textbf{and A4: enc-weakly-reflects-barb-set {success}}

\textbf{and A5: rel-reflects-divergence $T\text{Rel}$ Target} \textbf{and A6: enc-reflects-divergence}

\textbf{and A7: preorder $T\text{Rel}$}

\textbf{from A7 have A8: $T\text{Rel}^+ = T\text{Rel}$}

\textbf{using trancl-id[of $T\text{Rel}$]}

\textbf{unfolding preorder-on-def}

\textbf{by blast}

\textbf{from A7 have A9: $T\text{Rel}^+ = T\text{Rel}$}

\textbf{using refl-trancl[of $T\text{Rel}$] trancl-id[of $T\text{Rel}$]}

\textbf{unfolding preorder-on-def refl-on-def}

\textbf{by auto}

\textbf{define $Rel$ where $Rel = ind\text{RelRTPO}$ $T\text{Rel}$}

\textbf{hence B1: \forall S. (SourceTerm S, TargetTerm ([S])) \in Rel}

\textbf{by (simp add: ind\text{RelRTPO}.encR)}

\textbf{from Rel-def A8 have B2: $T\text{Rel} = \{(T1, T2), (TargetTerm T1, TargetTerm T2) \in Rel\}$}

\textbf{using ind\text{RelRTPO}-to-TRel(4)[where $T\text{Rel}=T\text{Rel}$]}

\textbf{by (auto simp add: ind\text{RelRTPO}.target)}

\textbf{from Rel-def A9 have B3: \forall S T. (SourceTerm S, TargetTerm T) \in Rel \rightarrow ([S], T) \in TRel}

\textbf{using ind\text{RelRTPO}-to-TRel(2)[where $T\text{Rel}=T\text{Rel}$]}

\textbf{trans-closure-of-$T\text{Rel}$-refl-cond[where $T\text{Rel}=T\text{Rel}$]}

\textbf{by simp}

\textbf{assume operational-complete $T\text{Rel}$ and weakly-operational-sound $T\text{Rel}$ and preorder $T\text{Rel}$}

\textbf{and weak-reduction-simulation $T\text{Rel}$ Target}

\& \forall P Q Q’. (P, Q) \in $T\text{Rel}$ \& Q \rightarrow Target* Q’

\rightarrow (\exists P’ Q”, P \rightarrow Target* P’’ \& Q’ \rightarrow Target* Q’’ \& (P”, Q”) \in $T\text{Rel}$)

\textbf{with Rel-def A8 A9 have B4: weak-reduction-correspondence-simulation $Rel$ ($STCal$ Source Target)}

\textbf{using WOC-iff-ind\text{RelRTPO}-is-reduction-correspondence-simulation[where $T\text{Rel}=T\text{Rel}$]}

\textbf{by simp}

\textbf{from Rel-def A7 have B5: preorder $Rel$}

265
using \textit{indRelRTPO-is-preorder}[\textbf{where} TRel=TRel]
unfolding preorder-on-def
by simp
from Rel-def A1 A2 A3 A4 have B6: rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) {success}
using enc-and-TRel-impl-indRelRTPO-weakly-respects-success[\textbf{where} TRel=TRel
and success=success]
by blast
from Rel-def A5 A6 have B7: rel-reflects-divergence Rel (STCal Source Target)
using enc-and-TRelimpl-indRelRTPO-reflect-divergence[\textbf{where} TRel=TRel]
by blast
show \(\exists\) Rel. \((\forall S. \text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}\)
\& TRel = \{(T1, T2). (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\}
\& \((\forall S T. \text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}\)
\& weak-reduction-correspondence-simulation Rel (STCal Source Target) \& preorder Rel
\& rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) \{success\}
\& rel-reflects-divergence Rel (STCal Source Target)
apply (rule exI) using B1 B2 B3 B4 B5 B6 B7 by blast
next
assume \(\exists\) Rel. \((\forall S. \text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}\)
\& TRel = \{(T1, T2). (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\}
\& \((\forall S T. \text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}\)
\& weak-reduction-correspondence-simulation Rel (STCal Source Target) \& preorder Rel
\& rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) \{success\}
\& rel-reflects-divergence Rel (STCal Source Target)
from this obtain Rel where C1: \(\forall S. \text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}\)
and C2: TRel = \{(T1, T2). (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\}
and C3: \(\forall S T. \text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}\)
and C4: weak-reduction-correspondence-simulation Rel (STCal Source Target)
and C5: preorder Rel and C6: rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) \{success\}
and C7: rel-reflects-divergence Rel (STCal Source Target)
by auto
from C1 C2 C3 C4 C5 have \(\exists\) Rel. \((\forall S. \text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}\)
\& TRel = \{(T1, T2). (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\}
\& \((\forall S T. \text{SourceTerm } S, \text{TargetTerm } T) \in \text{Rel} \rightarrow ([S], T) \in \text{TRel}\) \& preorder Rel
\& weak-reduction-correspondence-simulation Rel (STCal Source Target)
by blast
hence weakly-operational-corresponding TRel \& preorder TRel
\& weak-reduction-correspondence-simulation TRel Target
using WOC-wrt-preorder-iff-reduction-correspondence-simulation[\textbf{where} TRel=TRel]
by simp
moreover have \(\exists\) Rel. \((\forall S. \text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}\)
\& rel-weakly-respects-barb-set Rel (STCalWB SWB TWB) \{success\}
\& rel-reflects-divergence Rel (STCal Source Target)
apply (rule exI) using C1 C6 C7 by blast
hence enc-weakly-respects-barb-set \{success\} \& enc-reflects-divergence
using WSS-DR-iff-source-target-rel
by simp
moreover have rel-weakly-respects-barb-set TRel TWB \{success\}
proof auto
fix TP TQ TP'
assume \((TP, TQ) \in \text{TRel}\)
with C2 have \((\text{TargetTerm } TP, \text{TargetTerm } TQ) \in \text{Rel}\)
by simp
moreover assume \(TP \rightarrow (\text{Calculus TWB}) \ast TP’ \text{ and } TP’ \downarrow <TWB> \text{success}\)
hence \(\text{TargetTerm } TP \downarrow <\text{STCalWB SWB TWB}> \text{success}\)
using STCalWB-reachesBarbST
by blast
ultimately have \(\text{TargetTerm } TQ \downarrow <\text{STCalWB SWB TWB}> \text{success}\)
using C6
by blast
thus \(TQ \downarrow <TWB> \text{success}\)
using $\text{STCalWB-reachesBarbST}$
by blast

next
fix $TP \; TQ \; TQ'$
assume $(TP, \; TQ) \in TRel$
with simp

moreover assume $TQ \xrightarrow{\text{Calculus TWB}} TQ'$ and $TQ' \xrightarrow{\text{TWB}} \text{success}$
hence $\text{TargetTerm} \; TQ' \xleftarrow{\text{STCalWB SWB TWB}} \text{success}$
by blast

ultimately have $\text{TargetTerm} \; TP' \xleftarrow{\text{STCalWB SWB TWB}} \text{success}$
using $C6$
by blast
thus $TP' \xrightarrow{\text{TWB}} \text{success}$
by blast

qed

moreover from $C2 \; C7$ have $\text{rel-reflects-divergence} \; TRel \; \text{Target}$
using $\text{STCal-divergent}(2)$
by blast

ultimately
show $\text{weakly-operational-corresponding} \; TRel \land \text{preorder} \; TRel$
\land $\text{weak-reduction-bisimulation} \; TRel \; \text{Target}$
\land $\text{enc-weakly-respects-barb-set} \{\text{success}\}$ \land $\text{rel-weakly-respects-barb-set} \; TRel \; \text{TWB} \{\text{success}\}$
\land $\text{enc-reflects-divergence} \land \text{rel-reflects-divergence} \; TRel \; \text{Target}$
by fast

qed

lemma (in encoding-wrt-barbs) $\text{OC-SS-DR-wrt-preorder-iff-source-target-rel}$:
fixes $\text{success} :: \langle \text{barbs} \rangle$
and $TRel :: (\langle \text{procT} \times \text{procT} \rangle \; \text{set})$
shows $(\text{operational-corresponding} \; TRel \land \text{preorder} \; TRel \land \text{weak-reduction-bisimulation} \; TRel \; \text{Target}$
\land $\text{enc-weakly-respects-barb-set} \{\text{success}\}$
\land $\text{rel-weakly-respects-barb-set} \; TRel \; \text{TWB} \{\text{success}\}$
\land $\text{enc-reflects-divergence} \land \text{rel-reflects-divergence} \; TRel \; \text{Target}$
proof (rule iffI, (erule conjE)+)
assume $A1: \text{rel-weakly-preserves-barb-set} \; TRel \; \text{TWB} \{\text{success}\}$
and $A2: \text{rel-weakly-reflected-barb-set} \; TRel \; \text{TWB} \{\text{success}\}$
and $A3: \text{enc-weakly-preserves-barb-set} \{\text{success}\}$
and $A4: \text{enc-weakly-reflected-barb-set} \{\text{success}\}$
and $A5: \text{rel-reflected-divergence} \; TRel \; \text{Target}$ and $A6: \text{enc-reflected-divergence}$
and $A7: \text{preorder} \; TRel$
from $A7$ have $A8: \; TRel^* = TRel$
using trancl-id[of $TRel$]
unfolding preorder-on-def
by blast
from $A7$ have $A9: \; TRel^* = TRel$
using reflcl-trancl[of $TRel$], trancl-id[of $TRel$]
unfolding preorder-on-def refl-on-def
by auto
define $Rel$ where $Rel = \text{indRelRTPO} \; TRel$

hence $B1: \forall S. (\text{SourceTerm} \; S, \; \text{TargetTerm} \; ([S])) \in Rel$
by (simp add: indRelRTPO.encR)
from $\text{Rel-def} \; A8$ have $B2: \; TRel = \{ (T1, \; T2) . (\text{TargetTerm} \; T1, \; \text{TargetTerm} \; T2) \in \text{Rel} \}$
using \text{indRelRTPO-to-TRel}(4) |\text{where } \text{TRel} = \text{TRel}|
by (auto simp add: \text{indRelRTPO-target})
from \text{Rel-def A9} have B3: \text{\forall } S T . \text{(SourceTerm } S , \text{TargetTerm } T) \in \text{Rel} \longrightarrow ([S], T) \in \text{TRel}
using \text{indRelRTPO-to-TRel}(2) |\text{where } \text{TRel} = \text{TRel}|
\quad \text{trans-closure-of-TRel-refl-cond} |\text{where } \text{TRel} = \text{TRel}|
by simp
assume \text{operational-complete } \text{TRel} \text{ and operational-sound } \text{TRel} \text{ and preorder } \text{TRel}
and \text{weak-reduction-simulation } \text{TRel Target}
and \text{\forall } P Q Q' . \text{(P, Q) } \in \text{TRel} \land Q \longmapsto Target* Q' \longrightarrow (\exists P'. P \longmapsto Target* P' \land (P', Q') \in \text{TRel})
with \text{Rel-def A8 A9} have B4: \text{weak-reduction-bisimulation } \text{Rel (STCal Source Target)}
using \text{OC-iff-indRelRTPO-is-weak-reduction-bisimulation} |\text{where } \text{TRel} = \text{TRel}|
by simp
from \text{Rel-def A7} have B5: \text{preorder } \text{Rel}
using \text{indRelRTPO-is-preorder} |\text{where } \text{TRel} = \text{TRel}|
unfolding \text{preorder-on-def}
by simp
from \text{Rel-def A1 A2 A3 A4} have B6: \text{rel-weakly-respects-barb-set } \text{Rel (STCalWB SWB TWB)} \{ \text{success} \}
using \text{enc-and-TRel-impl-indRelRTPO-weakly-respects-success} |\text{where } \text{TRel} = \text{TRel}|
\quad \text{and }\text{success = success}|
by blast
from \text{Rel-def A5 A6} have B7: \text{rel-reflects-divergence } \text{Rel (STCal Source Target)}
using \text{enc-and-TRel-impl-indRelRTPO-reflect-divergence} |\text{where } \text{TRel} = \text{TRel}|
by blast
show \exists \text{Rel}. (\forall S . \text{(SourceTerm } S , \text{TargetTerm } ([S])) \in \text{Rel})
\land \text{TRel} = \{(T1, T2) . (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\}
\land (\forall S T . (\text{SourceTerm } S , \text{TargetTerm } T) \in \text{Rel} \longrightarrow ([S], T) \in \text{TRel})
\land \text{weak-reduction-bisimulation } \text{Rel (STCal Source Target)} \land \text{preorder Rel}
\land \text{rel-weakly-respects-barb-set } \text{Rel (STCalWB SWB TWB)} \{ \text{success} \}
\land \text{rel-reflects-divergence } \text{Rel (STCal Source Target)}
apply (rule exI) using B1 B2 B3 B4 B5 B6 B7 by blast
next
assume \exists \text{Rel}. (\forall S . \text{(SourceTerm } S , \text{TargetTerm } ([S])) \in \text{Rel})
\land \text{TRel} = \{(T1, T2) . (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\}
\land (\forall S T . (\text{SourceTerm } S , \text{TargetTerm } T) \in \text{Rel} \longrightarrow ([S], T) \in \text{TRel})
\land \text{weak-reduction-bisimulation } \text{Rel (STCal Source Target)} \land \text{preorder Rel}
\land \text{rel-weakly-respects-barb-set } \text{Rel (STCalWB SWB TWB)} \{ \text{success} \}
\land \text{rel-reflects-divergence } \text{Rel (STCal Source Target)}
from \text{this obtain } \text{Rel where C1: } \forall S . \text{(SourceTerm } S , \text{TargetTerm } ([S])) \in \text{Rel}
and C2: \text{TRel} = \{(T1, T2) . (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\}
and C3: \forall S T . (\text{SourceTerm } S , \text{TargetTerm } T) \in \text{Rel} \longrightarrow ([S], T) \in \text{TRel}
and C4: \text{weak-reduction-bisimulation } \text{Rel (STCal Source Target)} \land \text{preorder Rel}
and C5: \text{rel-weakly-respects-barb-set } \text{Rel (STCalWB SWB TWB)} \{ \text{success} \}
and C7: \text{rel-reflects-divergence } \text{Rel (STCal Source Target)}
by auto
from C1 C2 C3 C4 C5 have \exists \text{Rel}. (\forall S . \text{(SourceTerm } S , \text{TargetTerm } ([S])) \in \text{Rel})
\land \text{TRel} = \{(T1, T2) . (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\}
\land (\forall S T . (\text{SourceTerm } S , \text{TargetTerm } T) \in \text{Rel} \longrightarrow ([S], T) \in \text{TRel}) \land \text{preorder Rel}
\land \text{weak-reduction-bisimulation } \text{Rel (STCal Source Target)}
\land \text{rel-reflects-divergence } \text{Rel (STCal Source Target)}
by blast
\text{hence } \text{operational-corresponding } \text{TRel} \land \text{preorder } \text{TRel} \land \text{weak-reduction-bisimulation } \text{TRel Target}
using \text{OC-wrt-preorder-iff-weak-reduction-bisimulation} |\text{where } \text{TRel} = \text{TRel}|
by simp
\text{moreover have } \exists \text{Rel}. (\forall S . \text{(SourceTerm } S , \text{TargetTerm } ([S])) \in \text{Rel})
\land \text{rel-weakly-respects-barb-set } \text{Rel (STCalWB SWB TWB)} \{ \text{success} \}
\land \text{rel-reflects-divergence } \text{Rel (STCal Source Target)}
apply (rule exI) using C1 C6 C7 by blast
\text{hence } \text{enc-weakly-respects-barb-set } \{ \text{success} \} \land \text{enc-reflects-divergence}
using \text{WSS-DR-iff-source-target-rel}
by simp
\text{moreover have } \text{rel-weakly-respects-barb-set } \text{TRel TWB } \{ \text{success} \}
proof auto
fix TP TQ TP'  
assume (TP, TQ) ∈ TRel  
with C2 have (TargetTerm TP, TargetTerm TQ) ∈ Rel  
by simp  
moreover assume TP →→ (Calculus TWB)∗ TP' and TP'↓<TWB>success  
hence TargetTerm TP⇓<STCalWB SWB TWB>success  
using STCalWB-reachesBarbST  
by blast  
ultimately have TargetTerm TP⇓<STCalWB SWB TWB>success  
using C6  
by blast  
thus TP⇓<TWB>success  
using STCalWB-reachesBarbST  
by blast  
next  
fix TP TQ TQ'  
assume (TP, TQ) ∈ TRel  
with C2 have (TargetTerm TP, TargetTerm TQ) ∈ Rel  
by simp  
moreover assume TQ →→ (Calculus TWB)∗ TQ' and TQ'↓<TWB>success  
hence TargetTerm TQ⇓<STCalWB SWB TWB>success  
using STCalWB-reachesBarbST  
by blast  
ultimately have TargetTerm TQ⇓<STCalWB SWB TWB>success  
using C6  
by blast  
thus TQ⇓<TWB>success  
using STCalWB-reachesBarbST  
by blast  
qed  
moreover from C2 C7 have rel-reflects-divergence TRel Target  
using STCal-divergent(2)  
by blast  
ultimately  
show operational-corresponding TRel ∧ preorder TRel ∧ weak-reduction-bisimulation TRel Target  
∧ enc-weakly-respects-barb-set {success} ∨ rel-weakly-respects-barb-set TRel TWB {success}  
∧ enc-reflects-divergence ∨ rel-reflects-divergence TRel Target  
by fast  
qed  
lemma (in encoding-wrt-barbs) SOC-SS-DR-wrt-preorder-iff-source-target-rel:  
fixes success :: 'barbs  
and TRel :: ('procT × 'procT) set  
shows (strongly-operational-corresponding TRel ∧ preorder TRel  
∧ strong-reduction-bisimulation TRel Target  
∧ enc-respects-barb-set {success} ∨ rel-respects-barb-set TRel TWB {success}  
∧ enc-reflects-divergence ∨ rel-reflects-divergence TRel Target)  
= (∃ Rel. (∀ S, SourceTerm S, TargetTerm ([S])) ∈ Rel  
∧ TRel = {[T1, T2], (TargetTerm T1, TargetTerm T2) ∈ Rel}  
∧ (∀ S T, SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel  
∧ strong-reduction-bisimulation Rel (STCal Source Target) ∧ preorder Rel  
∧ rel-respects-barb-set Rel (STCalWB SWB TWB) {success}  
∧ rel-reflects-divergence Rel (STCal Source Target))  
proof (rule iffI, (erule conjE)+)  
assume A1: rel-preserves-barb-set TRel TWB {success}  
and A2: rel-reflects-barb-set TRel TWB {success}  
and A3: enc-preserves-barb-set {success} and A4: enc-reflects-barb-set {success}  
and A5: rel-reflects-divergence TRel Target and A6: enc-reflects-divergence  
and A7: preorder TRel  
from A7 have A8: TRel+ = TRel  
using trancl-id[of TRel]
unfolding preorder-on-def
by blast

from A7 have A9: TRel" = TRel
  using refl-trancl[of TRel] trancl-id[of TRel]
unfolding preorder-on-def refl-on-def
by auto

define Rel where Rel = indRelRTPO TRel

hence B1: ∀ S. (SourceTerm S, TargetTerm (⌈S⌉)) ∈ Rel
  by (simp add: indRelRTPO.encR)
from Rel-def A8 have B2: TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
  using indRelRTPO-to-TRel(4)[where TRel=TRel]
  by (auto simp add: indRelRTPO.target)
from Rel-def A9 have B3: ∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel
  using indRelRTPO-to-TRel(2)[where TRel=TRel]
trans-close-of-TRel-refl-cond[where TRel=TRel]
by simp

from Rel-def A7 have B5: preorder Rel
  using indRelRTPO-is-preorder[where TRel=TRel]
unfolding preorder-on-def
by simp

from Rel-def A1 A2 A3 A4 have B6: rel-respects-barb-set Rel (STCalWB SWB TWB) {success}
  using enc-and-TRel-impl-indRelRTPO-respects-success[where TRel=TRel and success=success]
  by blast
from Rel-def A5 A6 have B7: rel-reflects-divergence Rel (STCal Source Target)
  using enc-and-TRelimpl-indRelRTPO-reflect-divergence[where TRel=TRel]
  by blast

show ∃ Rel. (∀ S. (SourceTerm S, TargetTerm (⌈S⌉)) ∈ Rel)
  ∧ TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
  ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel)
  ∧ strong-reduction-bisimulation Rel (STCal Source Target) ∧ preorder Rel
  ∧ rel-respects-barb-set Rel (STCalWB SWB TWB) {success}
  ∧ rel-reflects-divergence Rel (STCal Source Target)
apply (rule exI) using B1 B2 B3 B4 B5 B6 B7 by blast

next

assume ∃ Rel. (∀ S. (SourceTerm S, TargetTerm (⌈S⌉)) ∈ Rel)
  ∧ TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
  ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel)
  ∧ strong-reduction-bisimulation Rel (STCal Source Target) ∧ preorder Rel
  ∧ rel-respects-barb-set Rel (STCalWB SWB TWB) {success}
  ∧ rel-reflects-divergence Rel (STCal Source Target)
from this obtain Rel where C1: ∀ S. (SourceTerm S, TargetTerm (⌈S⌉)) ∈ Rel
and C2: TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
and C3: ∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel
and C4: strong-reduction-bisimulation Rel (STCal Source Target) and C5: preorder Rel
and C6: rel-respects-barb-set Rel (STCalWB SWB TWB) {success}
and C7: rel-reflects-divergence Rel (STCal Source Target)
by auto

from C1 C2 C3 C4 C5 have ∃ Rel. (∀ S. (SourceTerm S, TargetTerm (⌈S⌉)) ∈ Rel)
  ∧ TRel = {(T1, T2). (TargetTerm T1, TargetTerm T2) ∈ Rel}
  ∧ (∀ S T. (SourceTerm S, TargetTerm T) ∈ Rel → ([S], T) ∈ TRel) ∧ preorder Rel
  ∧ strong-reduction-bisimulation Rel (STCal Source Target)
  by blast
hence strongly-operational-corresponding TRel ∧ preorder TRel
  ∧ strong-reduction-bisimulation TRel Target
  using SOC-wrt-preorder-iff-strong-reduction-bisimulation[where TRel=TRel]
by simp
moreover have \( \exists \text{Rel} \cdot (\forall S \cdot (\text{SourceTerm } S, \text{TargetTerm } ([S])) \in \text{Rel}) \)
\(\land \text{rel-respects-barb-set } \text{Rel} (\text{STCalWB } \text{SWB } \text{TWB}) \{\text{success}\}\)
\(\land \text{rel-reflects-divergence } \text{Rel} (\text{STCal Source Target})\)
apply (rule exI) using C1 C6 C7 by blast
hence enc-respects-barb-set \(\{\text{success}\}\) \(\land\) enc-reflects-divergence
using SS-DR-iff-source-target-rel
by simp
moreover have rel-respects-barb-set \text{TRel } \text{TWB } \{\text{success}\}
proof auto
fix TP TQ
assume \((TP, TQ) \in \text{TRel}\)
with C2 have \((\text{TargetTerm } TP, \text{TargetTerm } TQ) \in \text{Rel}\)
by simp
moreover assume \(TP\downarrow <\text{TWB}>\text{success}\)
hence TargetTerm \(TP\downarrow <\text{STCalWB } \text{SWB } \text{TWB}>\text{success}\)
using \(\text{STCalWB-hasBarbST}\)
by blast
ultimately have TargetTerm \(TQ\downarrow <\text{STCalWB } \text{SWB } \text{TWB}>\text{success}\)
using C6
by blast
thus \(TQ\downarrow <\text{TWB}>\text{success}\)
using \(\text{STCalWB-hasBarbST}\)
by blast
qed
moreover from C2 C7 have rel-reflects-divergence \text{TRel } \text{Target}
using STCal-divergent(2)
by blast
ultimately show strongly-operational-corresponding \text{TRel } \land \text{preorder } \text{TRel}\)
\(\land\) strong-reduction-bisimulation \text{TRel } \text{Target}\)
\(\land\) enc-respects-barb-set \(\{\text{success}\}\) \(\land\) rel-respects-barb-set \text{TRel } \text{TWB } \{\text{success}\}\)
\(\land\) enc-reflects-divergence \(\land\) rel-reflects-divergence \text{TRel } \text{Target}\)
by fast
qed

10.3 Full Abstraction and Operational Correspondence

To combine full abstraction and operational correspondence we consider a symmetric version of the
induced relation and assume that the relations SRel and TRel are equivalences. Then an encoding is
fully abstract w.r.t. SRel and TRel and operationally corresponding w.r.t. TRel such that TRel is a
bisimulation iff the induced relation contains both SRel and TRel and is a transitive bisimulation.

**Lemma** (in encoding) FS-OC-modulo-equivalences-iff-source-target-relation:
fixes SRel :: ('procS \times 'procS) set
and TRel :: ('procT \times 'procT) set
assumes eqS: equivalence SRel
\textbf{and} \texttt{eqT}: equivalence \texttt{TRel}

\textbf{shows} fully-abstract \texttt{SRel} \texttt{TRel}

\begin{itemize}
  \item operational-corresponding \texttt{TRel} \land weak-reduction-bisimulation \texttt{TRel} \texttt{Target}
  \end{itemize}

\texttt{iffI} \iff \exists \texttt{Rel}.

\begin{itemize}
  \item \(\forall S. (\texttt{SourceTerm} S, \texttt{TargetTerm} ([S])) \in \texttt{Rel} \land (\texttt{SourceTerm} S \in \texttt{Rel})\)
  \item \texttt{SRel} = \{(S1, S2), (\texttt{SourceTerm} S1, \texttt{SourceTerm} S2) \in \texttt{Rel}\}
  \item \texttt{TRel} = \{(T1, T2), (\texttt{TargetTerm} T1, \texttt{TargetTerm} T2) \in \texttt{Rel}\}
  \item \texttt{trans} \texttt{Rel} \land weak-reduction-bisimulation \texttt{Rel} (STCal Source Target)
\end{itemize}

\textbf{proof} (rule \texttt{iffI}, erule \texttt{conjE}, erule \texttt{conjE})

\textbf{assume} \texttt{A1}: fully-abstract \texttt{SRel} \texttt{TRel} \texttt{and} \texttt{A2}: operational-corresponding \texttt{TRel}

\textbf{and} \texttt{A3}: weak-reduction-bisimulation \texttt{TRel} \texttt{Target}

\textbf{from} \texttt{eqT} \textbf{have} \texttt{A4}: \texttt{TRel} = \texttt{TRel}

\textbf{using} reflc-trancl[of \texttt{TRel}] trancl-id[of \texttt{TRel}]

\textbf{unfolding} \texttt{equiv-def refl-on-def}

\textbf{by} \texttt{auto}

\textbf{have} \texttt{A5}:

\begin{itemize}
  \item \(\forall S. \texttt{SourceTerm} S \sim [\cdot] T < \texttt{TRel}> \texttt{TargetTerm} ([S]) \land \texttt{SourceTerm} S \in \texttt{Rel} \land (\texttt{TargetTerm} ([S]) \sim [\cdot] T < \texttt{TRel}> \texttt{SourceTerm} S \in \texttt{Rel})\)
\end{itemize}

\textbf{by} (simp add: \texttt{indRelTEQ_encR} \texttt{indRelTEQ_encL})

\textbf{moreover from} \texttt{A4} \textbf{have} \texttt{A6}: \texttt{TRel} = \{(T1, T2), \texttt{TargetTerm} T1 \sim [\cdot] T < \texttt{TRel}> \texttt{TargetTerm} T2\}

\textbf{using} \texttt{indRelTEQ-to-TRel}(\texttt{where} \texttt{TRel}=\texttt{TRel})

\textbf{trans-closure-of-TRel-refl-cond}[\texttt{where} \texttt{TRel}=\texttt{TRel}]

\textbf{by} (auto simp add: \texttt{indRelTEQ_target})

\textbf{moreover have} \texttt{A7}: \texttt{trans} (\texttt{indRelTEQ} \texttt{TRel})

\textbf{using} \texttt{indRelTEQ} \texttt{trans}[\texttt{where} \texttt{TRel}=\texttt{TRel}]

\textbf{unfolding} \texttt{trans-def}

\textbf{by} \texttt{blast}

\textbf{moreover have} \texttt{SRel} = \{(S1, S2), \texttt{SourceTerm} S1 \sim [\cdot] T < \texttt{TRel}> \texttt{SourceTerm} S2\}

\textbf{proof}

\textbf{from} \texttt{A6} \textbf{have} \(\forall S1 S2. ([S1], [S2]) \in \texttt{TRel} = \texttt{TargetTerm} ([S1]) \sim [\cdot] T < \texttt{TRel}> \texttt{TargetTerm} ([S2])\)

\textbf{by} \texttt{blast}

\textbf{moreover have} \texttt{indRelTEQ} \texttt{TRel} \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\} = \texttt{indRelTEQ} \texttt{TRel}

\textbf{by} (auto simp add: \texttt{indRelTEQ_encL})

\textbf{with} \texttt{A7} \textbf{have} \texttt{trans} (\texttt{indRelTEQ} \texttt{TRel} \cup \{(P, Q). \exists S. [S] \in T P \land S \in S Q\})

\textbf{unfolding} \texttt{trans-def}

\textbf{by} \texttt{blast}

\textbf{ultimately show} \texttt{SRel} = \{(S1, S2). \texttt{SourceTerm} S1 \sim [\cdot] T < \texttt{TRel}> \texttt{SourceTerm} S2\}

\textbf{using} \texttt{A1} \texttt{A5} full-abstraction-and-trans-relation-contains-TRel-impl-SRel[\texttt{where} \texttt{SRel}=\texttt{SRel} \texttt{and} \texttt{TRel}=\texttt{TRel} \texttt{and} \texttt{Rel}=\texttt{indRelTEQ} \texttt{TRel}]

\textbf{by} \texttt{blast}

\textbf{qed}

\textbf{moreover from} \texttt{eqT} \texttt{A2} \texttt{A3} \textbf{have} weak-reduction-bisimulation (\texttt{indRelTEQ} \texttt{TRel}) (STCal Source Target)

\textbf{using} \texttt{OC-wrt-equivalence-iff-indRelTEQ-weak-reduction-bisimulation}[\texttt{where} \texttt{TRel}=\texttt{TRel}]

\textbf{by} \texttt{blast}

\textbf{ultimately show} \(\exists \texttt{Rel} \land \texttt{SRel} = \texttt{TRel} \land \texttt{Rel} = \texttt{indRelTEQ} \texttt{TRel}\)

\textbf{by} \texttt{blast}

\textbf{next}

\textbf{assume} \texttt{SRel} = \{(S1, S2), \texttt{SourceTerm} S1, \texttt{SourceTerm} S2 \in \texttt{Rel}\}

\textbf{and} \texttt{TRel} = \{(T1, T2), \texttt{TargetTerm} T1, \texttt{TargetTerm} T2 \in \texttt{Rel}\}

\textbf{and} \texttt{trans} \texttt{Rel} \land weak-reduction-bisimulation \texttt{Rel} (STCal Source Target)

\textbf{by} \texttt{blast}

\textbf{from this obtain} \texttt{Rel} \texttt{where}

\textbf{B1}: \(\forall S. (\texttt{SourceTerm} S, \texttt{TargetTerm} ([S])) \in \texttt{Rel} \land (\texttt{TargetTerm} ([S]), \texttt{SourceTerm} S \in \texttt{Rel})\)

\textbf{and} \texttt{B2}: \texttt{SRel} = \{(S1, S2), (\texttt{SourceTerm} S1, \texttt{SourceTerm} S2) \in \texttt{Rel}\}

\textbf{and} \texttt{B3}: \texttt{TRel} = \{(T1, T2), (\texttt{TargetTerm} T1, \texttt{TargetTerm} T2) \in \texttt{Rel}\}

\textbf{and} \texttt{B4}: \texttt{trans} \texttt{Rel} \texttt{and} \texttt{B5}: weak-reduction-bisimulation \texttt{Rel} (STCal Source Target)

\textbf{by} \texttt{blast}
from B1 B2 B3 B4 have fully-abstract SRel TRel
using trans-source-target-relation-impl-fully-abstract[where Rel=Rel and SRel=SRel
and TRel=TRel]
by blast
moreover have operational-corresponding TRel \land weak-reduction-bisimulation TRel Target
proof
from eqT have C1: TRel^+ = TRel
using trancl-id[of TRel]
unfolding equiv-def
by blast
from eqT have C2: TRel^* = TRel
using refl-trancl[of TRel] trancl-id[of TRel]
unfolding equiv-def refl-on-def
by auto
from B1 have \forall S. (SourceTerm S, TargetTerm ([S])) \in Rel
by simp
moreover from B3 have \forall T1 T2. (T1, T2) \in TRel \longrightarrow (TargetTerm T1, TargetTerm T2) \in Rel
by simp
moreover from B3 C1 have \forall T1 T2. (TargetTerm T1, TargetTerm T2) \in Rel \longrightarrow (T1, T2) \in TRel^+
by simp
moreover have \forall S T. (SourceTerm S, TargetTerm T) \in Rel \longrightarrow ([S], T) \in TRel^*
proof clarify
fix S T
from B1 have (TargetTerm ([S]), SourceTerm S) \in Rel
by simp
moreover assume (SourceTerm S, TargetTerm T) \in Rel
ultimately have (TargetTerm ([S]), TargetTerm T) \in Rel
using B4
unfolding trans-def
by blast
with B3 C2 show ([S], T) \in TRel^*
by simp
qed
ultimately have \exists Rel. (\forall S. (SourceTerm S, TargetTerm ([S])) \in Rel)
\land (\forall T1 T2. (T1, T2) \in TRel \longrightarrow (TargetTerm T1, TargetTerm T2) \in Rel)
\land (\forall T1 T2. (TargetTerm T1, TargetTerm T2) \in Rel \longrightarrow (T1, T2) \in TRel^+)
\land (\forall S T. (SourceTerm S, TargetTerm T) \in Rel \longrightarrow ([S], T) \in TRel^*)
\land weak-reduction-bisimulation Rel (STCal Source Target)
using B3
by blast
with C1 C2 show operational-corresponding TRel \land weak-reduction-bisimulation TRel Target
using OC-iff-weak-reduction-bisimulation[where TRel=TRel]
by auto
qed
ultimately show fully-abstract SRel TRel \land operational-corresponding TRel
\land weak-reduction-bisimulation TRel Target
by simp
qed

lemma (in encoding) FA-SOC-modulo-equivalences-iff-source-target-relation:
fixes SRel :: ('procS \times 'procS) set
and TRel :: ('procT \times 'procT) set
assumes eqS: equivalence SRel
and eqT: equivalence TRel
shows fully-abstract SRel TRel \land strongly-operational-corresponding TRel
\land strong-reduction-bisimulation TRel Target \iff (\exists Rel.
(\forall S. (SourceTerm S, TargetTerm ([S])) \in Rel \land (TargetTerm ([S]), SourceTerm S) \in Rel)
\land SRel = \{(S1, S2), (SourceTerm S1, SourceTerm S2) \in Rel\}
\land TRel = \{(T1, T2), (TargetTerm T1, TargetTerm T2) \in Rel\} \land trans Rel
\land strong-reduction-bisimulation Rel (STCal Source Target))
proof (rule iffI, erule conjE, erule conjE)
assume \( A1 \): fully-abstract \( SRel \) \( TRel \) and \( A2 \): strongly-operational-corresponding \( TRel \)
and \( A3 \): strong-reduction-bisimulation \( TRel \) \( Target \)
from \( eqT \) have \( A4 \): \( TRel^* = TRel \)
  using refl-trancl[of \( TRel \)] trancl-id[of \( TRel \)]
unfolding equiv-def refl-on-def
by auto
have \( A5 \):
\[
\forall S. \text{SourceTerm } S \sim^{\downarrow} \{ T \in T<TRel> \} \text{ TargetTerm } (\{ S \}) \land \text{TargetTerm } (\{ S \}) \sim^{\downarrow} \{ T \in T<TRel> \} \text{ SourceTerm } S
\]
by (simp add: \( \text{indRelTEQ.encR \( \text{encR} \) \( \text{indRelTEQ.encL} \) }\))
moreover from \( A4 \) have \( A6 \): \( TRel = \{(T1, T2). \text{TargetTerm } T1 \sim^{\downarrow} \{ T \in T<TRel> \} \text{ TargetTerm } T2\} \)
  using \( \text{indRelTEQ.to-TRel(4)} \)\{where \( \text{TRel=TRel}\)\}
  trans-closure-of-TRel-refl-cond\{where \( \text{TRel=TRel}\)\}
by (auto simp add: \( \text{indRelTEQ.target} \) )
moreover have \( A7 \): \( \text{trans } (\text{indRelTEQ } TRel) \)
  using \( \text{indRelTEQ.tran} \)\{where \( \text{TRel=TRel}\)\}
unfolding tran-def
by blast
moreover have \( SRel = \{(S1, S2). \text{SourceTerm } S1 \sim^{\downarrow} \{ T \in T<TRel> \} \text{ SourceTerm } S2\} \)
proof –
from \( A6 \) have \( \forall S1 S2. (\{ S1 \}, \{ S2 \}) \in TRel = \\text{TargetTerm } (\{ S1 \}) \sim^{\downarrow} \{ T \in T<TRel> \} \text{ TargetTerm } (\{ S2 \}) \)
by blast
moreover have \( \text{indRelTEQ } TRel \cup \{(P, Q). \exists S. (S) \in T P \land S \in S Q\} = \text{indRelTEQ } TRel \)
by (auto simp add: \( \text{indRelTEQ.encL} \) )
with \( A7 \) have \( \text{trans } (\text{indRelTEQ } TRel \cup \{(P, Q). \exists S. (S) \in T P \land S \in S Q\}) \)
unfolding tran-def
by blast
ultimately show \( SRel = \{(S1, S2). \text{SourceTerm } S1 \sim^{\downarrow} \{ T \in T<TRel> \} \text{ SourceTerm } S2\} \)
  using \( A1 \ A5 \text{ full-abstraction-and-trans-relation-contains-TRel-impl-SRel} \)\{where \( \text{TRel=TRel}\)\}
  \( \text{SRel=SRel} \) \( \text{and} \) \( \text{TRel=TRel} \) \( \text{and} \) \( \text{Rel=indRelTEQ } TRel\)
by blast
qed
moreover from \( eqT \) \( A2 \) \( A3 \) have \( \text{strong-reduction-bisimulation } (\text{indRelTEQ TRel}) \) \( \text{(STCal Source Target)} \)
  using \( \text{SOC-wrt-equivalence-iff-indRelTEQ-strong-reduction-bisimulation} \)\{where \( \text{TRel=TRel}\)\}
by blast
ultimately show \( \exists \text{Rel. } (\forall S. (\text{SourceTerm } S, \text{TargetTerm } (\{ S \})) \in \text{Rel } \land (\text{TargetTerm } (\{ S \}), \text{SourceTerm } S) \in \text{Rel} \)
\( \land \text{SRel} = \{(S1, S2), (\text{SourceTerm } S1, \text{SourceTerm } S2) \in \text{Rel}\} \)
\( \land \text{TRel} = \{(T1, T2), (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\} \land \text{trans Rel} \)
\( \land \text{strong-reduction-bisimulation Rel } (\text{STCal Source Target}) \)
by blast
next
assume \( \exists \text{Rel. } (\forall S. (\text{SourceTerm } S, \text{TargetTerm } (\{ S \})) \in \text{Rel } \land (\text{TargetTerm } (\{ S \}), \text{SourceTerm } S) \in \text{Rel} \)
\( \land \text{SRel} = \{(S1, S2), (\text{SourceTerm } S1, \text{SourceTerm } S2) \in \text{Rel}\} \)
\( \land \text{TRel} = \{(T1, T2), (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\} \land \text{trans Rel} \)
\( \land \text{strong-reduction-bisimulation Rel } (\text{STCal Source Target}) \)
from this obtain \( \text{Rel where} \)
\( B1: \forall S. (\text{SourceTerm } S, \text{TargetTerm } (\{ S \})) \in \text{Rel } \land (\text{TargetTerm } (\{ S \}), \text{SourceTerm } S) \in \text{Rel} \)
and \( B2: \text{SRel} = \{(S1, S2), (\text{SourceTerm } S1, \text{SourceTerm } S2) \in \text{Rel}\} \)
and \( B3: \text{TRel} = \{(T1, T2), (\text{TargetTerm } T1, \text{TargetTerm } T2) \in \text{Rel}\} \) \( \text{and} \) \( B4: \text{trans Rel} \)
and \( B5: \text{strong-reduction-bisimulation Rel } (\text{STCal Source Target}) \)
by blast
from \( B1 \) \( B2 \) \( B3 \) \( B4 \) have \( \text{fully-abstract } SRel \) \( TRel \)
  using \( \text{trans-source-target-relation-impl-fully-abstract} \)\{where \( \text{Rel=Rel} \) \( \text{and} \) \( \text{SRel=SRel} \) \( \text{and} \) \( \text{TRel=TRel} \)\}
by blast
moreover
have \( \text{strongly-operational-corresponding } TRel \land \text{strong-reduction-bisimulation } TRel \) \( \text{Target} \)
proof –
from \( eqT \) have \( C1: TRel^+ = TRel \)
  using trancl-id[of \( TRel \)]
unfolding equiv-def refl-on-def
by blast
from eqT have C2: TRel$^* =$ TRel
    using refl-trancl[of TRel] trancl-id[of TRel]
unfolding equiv-def refl-on-def
by auto
from B1 have $\forall$ S. (SourceTerm S, TargetTerm ([S])) $\in$ Rel
    by simp
moreover from B3 have $\forall$ T1 T2. (T1, T2) $\in$ TRel $\implies$ (TargetTerm T1, TargetTerm T2) $\in$ Rel
    by simp
moreover from B3 C1
have $\forall$ T1 T2. (TargetTerm T1, TargetTerm T2) $\in$ Rel $\implies$ (T1, T2) $\in$ TRel$^+$
    by simp
moreover have $\forall$ S T. (SourceTerm S, TargetTerm T) $\in$ Rel $\implies$ ([S], T) $\in$ TRel$^*$
proof clarify
fix S T
from B1 have (TargetTerm ([S]), SourceTerm S) $\in$ Rel
    by simp
moreover assume (SourceTerm S, TargetTerm T) $\in$ Rel
ultimately have (TargetTerm ([S]), TargetTerm T) $\in$ Rel
using B4
unfolding trans-def
by blast
with B3 C2 show ([S], T) $\in$ TRel$^*$
    by simp
qed
ultimately have $\exists$ Rel. ($\forall$ S. (SourceTerm S, TargetTerm ([S])) $\in$ Rel)
    $\land$ ($\forall$ T1 T2. (T1, T2) $\in$ TRel $\implies$ (TargetTerm T1, TargetTerm T2) $\in$ Rel)
    $\land$ ($\forall$ T1 T2. (TargetTerm T1, TargetTerm T2) $\in$ Rel $\implies$ (T1, T2) $\in$ TRel$^+$)
    $\land$ ($\forall$ S T. (SourceTerm S, TargetTerm T) $\in$ Rel $\implies$ ([S], T) $\in$ TRel$^*$)
    $\land$ strong-reduction-bisimulation Rel (STCal Source Target)
    using B3
    by blast
with C1 C2
show strongly-operational-corresponding TRel $\land$ strong-reduction-bisimulation TRel Target
    using SOC-iff-strong-reduction-bisimulation[where TRel=TRel]
    by auto
qed
ultimately show fully-abstract SRel TRel $\land$ strongly-operational-corresponding TRel
    $\land$ strong-reduction-bisimulation TRel Target
    by simp
qed

An encoding that is fully abstract w.r.t. the equivalences SRel and TRel and operationally corresponding w.r.t. TRel ensures that SRel is a bisimulation iff TRel is a bisimulation.

lemma (in encoding) FA-and-OC-and-TRel-impl-SRel-bisimulation:
fixes SRel :: ('procS × 'procS) set
and TRel :: ('procT × 'procT) set
assumes fullAbs: fully-abstract SRel TRel
    and opCom: operational-complete TRel
and opSou: operational-sound TRel
and symmT: sym TRel
and transT: trans TRel
and bisimT: weak-reduction-bisimulation TRel Target
shows weak-reduction-bisimulation SRel Source
proof auto
fix SP SQ SP'
assume SP' $\mapsto$ Source$^*$ SP'
with opCom obtain TP' where A1: [SP] $\mapsto$ Target$^*$ TP' and A2: ([SP'], TP') $\in$ TRel
    by blast

275
assume \((SP, SQ) \in SRel\)
with \(\text{fullAbs have \([\{SP\}, \{SQ\}] \in TRel\)}\)
by simp
with \(\text{bisimT \ A1 obtain \(TQ'\) where \(A3: [SQ] \rightarrow \text{Target} \ TQ'\) and \(A4: (TP', TQ') \in TRel\)}\)
by blast
from \(\text{A2 opSou obtain \(SQ'\) where \(A5: \text{Source} \ SQ'\) and \(A6: ([SP'], [TQ']) \in TRel\)}\)
by blast
unfolding \(\text{trans-def sym-def}\)
by blast
with \(\text{fullAbs \ A5 show \(\exists SQ'. \ SQ \rightarrow \text{Source} \ SQ' \land (SP', SQ') \in SRel\)}\)
by blast
qed

lemma (in encoding) \(\text{FA-and-SOC-and-TRel-impl-SRel-strong-bisimulation}\):
fixes \(SRel:: ('\text{procS} \times '\text{procS})\) set
and \(TRel:: ('\text{procT} \times '\text{procT})\) set
assumes \(\text{fullAbs: fully-abstract SRel TRel}\)
and \(\text{opCom: strongly-operational-complete TRel}\)
and \(\text{opSou: strongly-operational-sound TRel}\)
and \(\text{symmT: \sym\ TRel}\)
and \(\text{transT: \trans\ TRel}\)
and \(\text{bisimT: strong-reduction-bisimulation TRel Target}\)
shows \(\text{strong-reduction-bisimulation SRel Source}\)
proof auto
fix \(SP SQ SP'\)
assume \(\text{SP \rightarrow Source} SP'\)
with \(\text{opCom obtain \(TP'\) where \(A1: [SP] \rightarrow \text{Target} \ TP'\) and \(A2: ([SP'], TP') \in TRel\)}\)
by blast
assume \(\text{(SP, SQ) \in SRel}\)
with \(\text{fullAbs have \([\{SP\}, \{SQ\}] \in TRel\)}\)
by simp
with \(\text{bisimT \ A1 obtain \(TQ'\) where \(A3: [SQ] \rightarrow \text{Target} \ TQ'\) and \(A4: (TP', TQ') \in TRel\)}\)
by blast
from \(\text{A2 opSou obtain \(SQ'\) where \(A5: \text{Source} \ SQ'\) and \(A6: ([SP'], [TQ']) \in TRel\)}\)
by blast
unfolding \(\text{trans-def sym-def}\)
by blast
with \(\text{fullAbs \ A5 show \(\exists SQ'. \ SQ \rightarrow \text{Source} \ SQ' \land (SP', SQ') \in SRel\)}\)
by blast
next
fix \(SP SQ SQ'\)
assume \(\text{SQ \rightarrow Source} SQ'\)
with opCom obtain $TQ'$ where $B1$: $\llbracket SQ \rrbracket \rightarrow Target \ TQ'$ and $B2$: $\llbracket SQ' \rrbracket$, $TQ' \in TRel$
by blast
assume $(SP, SQ) \in SRel$
with fullAbs have $\llbracket SP \rrbracket$, $\llbracket SQ \rrbracket \in TRel$
by simp
with bisimT $B1$ obtain $TP'$ where $B3$: $\llbracket SP \rrbracket \rightarrow Target \ TP'$ and $B4$: $(TP', TQ') \in TRel$
by blast
from $B3$ opSou obtain $SP'$ where $B5$: $SP \rightarrow Source \ SP'$ and $B6$: $\llbracket SP \rrbracket$, $TP' \in TRel$
by blast
unfolding trans-def sym-def
by blast
with fullAbs $B5$ show $\exists SP'$. $SP \rightarrow Source \ SP' \land (SP', SQ') \in SRel$
by blast
qed

lemma (in encoding) FA-and-OC-impl-SRel-iff-TRel-bisimulation:
fixes $SRel$ :: $(\text{proc} S \times \text{proc} S)$ set
and $TRel$ :: $(\text{proc} T \times \text{proc} T)$ set
assumes fullAbs: fully-abstract $SRel \ TRel$
and opCor: operational-corresponding $TRel$
and symmT: $\text{sym} \ TRel$
and transT: trans $TRel$
and surj: $\forall T, \exists S. \ T = \llbracket S \rrbracket$
shows weak-reduction-bisimulation $SRel \ Source \rightarrow weak-reduction-bisimulation \ TRel \ Target$
proof
assume bisimS: weak-reduction-bisimulation $SRel \ Source$
have weak-reduction-simulation $TRel \ Target$
proof clarify
fix $TP \ TQ \ TP'$
from surj have $\exists S$. $TP = \llbracket S \rrbracket$
by simp
from this obtain $SP$ where $A1$: $\llbracket SP \rrbracket = TP$
by simp
from surj have $\exists S$. $TQ = \llbracket S \rrbracket$
by simp
from this obtain $SQ$ where $A2$: $\llbracket SQ \rrbracket = TQ$
by simp
assume $TP \rightarrow Target* \ TP'$
with opCor $A1$ obtain $SP'$ where $A3$: $SP \rightarrow Source* \ SP'$ and $A4$: $\llbracket SP' \rrbracket$, $TP' \in TRel$
by blast
assume $(TP, TQ) \in TRel$
with fullAbs $A1 \ A2$ have $(SP, SQ) \in SRel$
by simp
with bisimS $A3$ obtain $SQ'$ where $A5$: $SQ \rightarrow Source* \ SQ'$ and $A6$: $(SP', SQ') \in SRel$
by blast
from opCor $A2 \ A5$ obtain $TQ'$ where $A7$: $TQ \rightarrow Target* \ TQ'$ and $A8$: $\llbracket SQ' \rrbracket$, $TQ' \in TRel$
by blast
from symmT $A4$ have $(TP', [SP']) \in TRel$
unfolding sym-def
by simp
moreover from fullAbs $A6$ have $\llbracket SP' \rrbracket$, $\llbracket SQ' \rrbracket \in TRel$
by simp
ultimately have $(TP', TQ') \in TRel$
using transT $A8$
unfolding trans-def
by blast
with $A7$ show $\exists TQ'$. $TQ \rightarrow Target* \ TQ' \land (TP', TQ') \in TRel$
by blast
qed
with symmT show weak-reduction-bisimulation $TRel \ Target$

277
using symm-weak-reduction-simulation-is-bisimulation[where Rel = TRel and Cal = Target]
by blast
next
assume weak-reduction-bisimulation TRel Target
with fullAbs opCor symmT transT show weak-reduction-bisimulation SRel Source
using FA-and-OC-and-TRel-impl-SRel-bisimulation[where SRel = SRel and TRel = TRel]
by blast
qed
end