A Comprehensive Framework for Saturation Theorem Proving

Sophie Tourret

March 19, 2025

Abstract

This Isabelle/HOL formalization is the companion of the technical report "A comprehensive framework for saturation theorem proving", itself companion of the eponym IJCAR 2020 paper, written by Uwe Waldmann, Sophie Tourret, Simon Robillard and Jasmin Blanchette. It verifies a framework for formal refutational completeness proofs of abstract provers that implement saturation calculi, such as ordered resolution or superposition, and allows to model entire prover architectures in such a way that the static refutational completeness of a calculus immediately implies the dynamic refutational completeness of a prover implementing the calculus using a variant of the given clause loop.

The technical report "A comprehensive framework for saturation theorem proving" is available at http://matryoshka.gforge.inria.fr/pubs/satur_report.pdf. The names of the Isabelle lemmas and theorems corresponding to the results in the report are indicated in the margin of the report.

Contents

1 Calculi Based on a Redundancy Criterion

This section introduces the most basic notions upon which the framework is built: consequence relations and inference systems. It also defines the notion of a family of consequence relations and of redundancy criteria. This corresponds to sections 2.1 and 2.2 of the report.

theory *Calculus*
  imports
    Ordered-Resolution-Prover.Lazy-List-Liminf
    Ordered-Resolution-Prover.Lazy-List-Chain
begin

1.1 Consequence Relations

locale *consequence-relation* =
  fixes
    $Bot :: 'f\ set$ **and**
    $entails :: 'f\ set \Rightarrow 'f\ set \Rightarrow bool$ (**infix** ‹⊨› *50*)
  assumes
    bot-not-empty: $Bot \neq \{\}$ **and**
    bot-entails-all: $B \in Bot \Longrightarrow \{B\} \models N1$ **and**
    subset-entailed: $N2 \subseteq N1 \Longrightarrow N1 \models N2$ **and**
    all-formulas-entailed: $(\forall C \in N2.\ N1 \models \{C\}) \Longrightarrow N1 \models N2$ **and**
    entails-trans[*trans*]: $N1 \models N2 \Longrightarrow N2 \models N3 \Longrightarrow N1 \models N3$
begin

lemma *entail-set-all-formulas*: $N1 \models N2 \longleftrightarrow (\forall C \in N2.\ N1 \models \{C\})$
  ⟨*proof*⟩

lemma *entail-union*: $N \models N1 \wedge N \models N2 \longleftrightarrow N \models N1 \cup N2$
  ⟨*proof*⟩

lemma *entail-unions*: $(\forall i \in I.\ N \models Ni\ i) \longleftrightarrow N \models \bigcup\ (Ni\ `\ I)$
  ⟨*proof*⟩

lemma *entail-all-bot*: $(\exists B \in Bot.\ N \models \{B\}) \Longrightarrow \forall B' \in Bot.\ N \models \{B'\}$
  ⟨*proof*⟩

lemma *entails-trans-strong*: $N1 \models N2 \Longrightarrow N1 \cup N2 \models N3 \Longrightarrow N1 \models N3$
  ⟨*proof*⟩

end

1.2 Families of Consequence Relations

locale *consequence-relation-family* =
  fixes
    $Bot :: 'f\ set$ **and**
    $Q :: 'q\ set$ **and**
    $entails\text{-}q :: 'q \Rightarrow 'f\ set \Rightarrow 'f\ set \Rightarrow bool$

assumes
Q-nonempty: $Q \neq \{\}$ **and**
q-cons-rel: $\forall\, q \in Q.$ *consequence-relation Bot* (*entails-q q*)
begin

lemma *bot-not-empty*: $Bot \neq \{\}$
⟨*proof*⟩

definition *entails* :: $'f$ *set* ⇒ $'f$ *set* ⇒ *bool* (**infix** ‹$\models Q$› *50*) **where**
$N1 \models Q\ N2 \longleftrightarrow (\forall\, q \in Q.\ entails\text{-}q\ q\ N1\ N2)$

lemma *intersect-cons-rel-family*: *consequence-relation Bot entails*
⟨*proof*⟩

end

1.3 Inference Systems

datatype $'f$ *inference* =
Infer (*prems-of*: $'f$ *list*) (*concl-of*: $'f$)

locale *inference-system* =
fixes
Inf :: ‹$'f$ *inference set*›
begin

definition *Inf-from* :: $'f$ *set* ⇒ $'f$ *inference set* **where**
$Inf\text{-}from\ N = \{\iota \in Inf.\ set\ (prems\text{-}of\ \iota) \subseteq N\}$

definition *Inf-between* :: $'f$ *set* ⇒ $'f$ *set* ⇒ $'f$ *inference set* **where**
$Inf\text{-}between\ N\ M = Inf\text{-}from\ (N \cup M) - Inf\text{-}from\ (N - M)$

lemma *Inf-if-Inf-from*: $\iota \in Inf\text{-}from\ N \Longrightarrow \iota \in Inf$
⟨*proof*⟩

lemma *Inf-if-Inf-between*: $\iota \in Inf\text{-}between\ N\ M \Longrightarrow \iota \in Inf$
⟨*proof*⟩

lemma *Inf-between-alt*:
$Inf\text{-}between\ N\ M = \{\iota \in Inf.\ \iota \in Inf\text{-}from\ (N \cup M) \wedge set\ (prems\text{-}of\ \iota) \cap M \neq \{\}\}$
⟨*proof*⟩

lemma *Inf-from-mono*: $N \subseteq N' \Longrightarrow Inf\text{-}from\ N \subseteq Inf\text{-}from\ N'$
⟨*proof*⟩

lemma *Inf-between-mono*: $N \subseteq N' \Longrightarrow M \subseteq M' \Longrightarrow Inf\text{-}between\ N\ M \subseteq Inf\text{-}between\ N'\ M'$
⟨*proof*⟩

end

1.4 Families of Inference Systems

locale *inference-system-family* =
fixes
Q :: $'q$ *set* **and**

$Inf\text{-}q :: 'q \Rightarrow 'f\ inference\ set$
assumes
$Q\text{-}nonempty$: $Q \neq \{\}$
begin

definition $Inf\text{-}from\text{-}q :: 'q \Rightarrow 'f\ set \Rightarrow 'f\ inference\ set$ **where**
$Inf\text{-}from\text{-}q\ q = inference\text{-}system.Inf\text{-}from\ (Inf\text{-}q\ q)$

definition $Inf\text{-}between\text{-}q :: 'q \Rightarrow 'f\ set \Rightarrow 'f\ set \Rightarrow 'f\ inference\ set$ **where**
$Inf\text{-}between\text{-}q\ q = inference\text{-}system.Inf\text{-}between\ (Inf\text{-}q\ q)$

lemma $Inf\text{-}between\text{-}q\text{-}alt$:
$Inf\text{-}between\text{-}q\ q\ N\ M = \{\iota \in Inf\text{-}q\ q.\ \iota \in Inf\text{-}from\text{-}q\ q\ (N \cup M) \wedge set\ (prems\text{-}of\ \iota) \cap M \neq \{\}\}$
$\langle proof \rangle$

end

1.5 Calculi Based on a Single Redundancy Criterion

locale $calculus = inference\text{-}system\ Inf + consequence\text{-}relation\ Bot\ entails$
for
$Bot :: 'f\ set$ **and**
$Inf :: \langle 'f\ inference\ set \rangle$ **and**
$entails :: 'f\ set \Rightarrow 'f\ set \Rightarrow bool$ (**infix** $\langle \models \rangle$ 50)
+ **fixes**
$Red\text{-}I :: 'f\ set \Rightarrow 'f\ inference\ set$ **and**
$Red\text{-}F :: 'f\ set \Rightarrow 'f\ set$
assumes
$Red\text{-}I\text{-}to\text{-}Inf$: $Red\text{-}I\ N \subseteq Inf$ **and**
$Red\text{-}F\text{-}Bot$: $B \in Bot \Longrightarrow N \models \{B\} \Longrightarrow N - Red\text{-}F\ N \models \{B\}$ **and**
$Red\text{-}F\text{-}of\text{-}subset$: $N \subseteq N' \Longrightarrow Red\text{-}F\ N \subseteq Red\text{-}F\ N'$ **and**
$Red\text{-}I\text{-}of\text{-}subset$: $N \subseteq N' \Longrightarrow Red\text{-}I\ N \subseteq Red\text{-}I\ N'$ **and**
$Red\text{-}F\text{-}of\text{-}Red\text{-}F\text{-}subset$: $N' \subseteq Red\text{-}F\ N \Longrightarrow Red\text{-}F\ N \subseteq Red\text{-}F\ (N - N')$ **and**
$Red\text{-}I\text{-}of\text{-}Red\text{-}F\text{-}subset$: $N' \subseteq Red\text{-}F\ N \Longrightarrow Red\text{-}I\ N \subseteq Red\text{-}I\ (N - N')$ **and**
$Red\text{-}I\text{-}of\text{-}Inf\text{-}to\text{-}N$: $\iota \in Inf \Longrightarrow concl\text{-}of\ \iota \in N \Longrightarrow \iota \in Red\text{-}I\ N$
begin

lemma $Red\text{-}I\text{-}of\text{-}Inf\text{-}to\text{-}N\text{-}subset$: $\{\iota \in Inf.\ concl\text{-}of\ \iota \in N\} \subseteq Red\text{-}I\ N$
$\langle proof \rangle$

lemma $red\text{-}concl\text{-}to\text{-}red\text{-}inf$:
assumes
$i\text{-}in$: $\iota \in Inf$ **and**
$concl$: $concl\text{-}of\ \iota \in Red\text{-}F\ N$
shows $\iota \in Red\text{-}I\ N$
$\langle proof \rangle$

definition $saturated :: 'f\ set \Rightarrow bool$ **where**
$saturated\ N \longleftrightarrow Inf\text{-}from\ N \subseteq Red\text{-}I\ N$

definition $reduc\text{-}saturated :: 'f\ set \Rightarrow bool$ **where**
$reduc\text{-}saturated\ N \longleftrightarrow Inf\text{-}from\ (N - Red\text{-}F\ N) \subseteq Red\text{-}I\ N$

lemma $Red\text{-}I\text{-}without\text{-}red\text{-}F$:
$Red\text{-}I\ (N - Red\text{-}F\ N) = Red\text{-}I\ N$

⟨*proof*⟩

lemma *saturated-without-red-F*:
  assumes *saturated*: *saturated N*
  shows *saturated* (*N* − *Red-F N*)
⟨*proof*⟩

definition *fair* :: *'f set llist* ⇒ *bool* **where**
  fair Ns ⟷ *Inf-from* (*Liminf-llist Ns*) ⊆ *Sup-llist* (*lmap Red-I Ns*)

inductive *derive* :: *'f set* ⇒ *'f set* ⇒ *bool* (**infix** ‹▷› *50*) **where**
  derive: *M* − *N* ⊆ *Red-F N* ⟹ *M* ▷ *N*

lemma *gt-Max-notin*: ‹*finite A* ⟹ *A* ≠ {} ⟹ *x* > *Max A* ⟹ *x* ∉ *A*› ⟨*proof*⟩

lemma *equiv-Sup-Liminf*:
  assumes
    in-Sup: *C* ∈ *Sup-llist Ns* **and**
    not-in-Liminf: *C* ∉ *Liminf-llist Ns*
  shows
    ∃ *i* ∈ {*i*. *enat* (*Suc i*) < *llength Ns*}. *C* ∈ *lnth Ns i* − *lnth Ns* (*Suc i*)
⟨*proof*⟩

lemma *Red-in-Sup*:
  assumes *deriv*: *chain* (▷) *Ns*
  shows *Sup-llist Ns* − *Liminf-llist Ns* ⊆ *Red-F* (*Sup-llist Ns*)
⟨*proof*⟩

lemma *Red-I-subset-Liminf*:
  assumes *deriv*: ‹*chain* (▷) *Ns*› **and**
    i: ‹*enat i* < *llength Ns*›
  shows ‹*Red-I* (*lnth Ns i*) ⊆ *Red-I* (*Liminf-llist Ns*)›
⟨*proof*⟩

lemma *Red-F-subset-Liminf*:
  assumes *deriv*: ‹*chain* (▷) *Ns*› **and**
    i: ‹*enat i* < *llength Ns*›
  shows ‹*Red-F* (*lnth Ns i*) ⊆ *Red-F* (*Liminf-llist Ns*)›
⟨*proof*⟩

lemma *i-in-Liminf-or-Red-F*:
  assumes
    deriv: ‹*chain* (▷) *Ns*› **and**
    i: ‹*enat i* < *llength Ns*›
  shows ‹*lnth Ns i* ⊆ *Red-F* (*Liminf-llist Ns*) ∪ *Liminf-llist Ns*›
⟨*proof*⟩

lemma *fair-implies-Liminf-saturated*:
  assumes
    deriv: ‹*chain* (▷) *Ns*› **and**

fair: ‹*fair Ns*›
shows ‹*saturated* (*Liminf-llist Ns*)›
⟨*proof*⟩

end

locale *statically-complete-calculus* = *calculus* +
assumes *statically-complete*: $B \in Bot \Longrightarrow$ *saturated* $N \Longrightarrow N \models \{B\} \Longrightarrow \exists B' \in Bot.\ B' \in N$
begin

lemma *dynamically-complete-Liminf*:
fixes *B Ns*
assumes
bot-elem: ‹$B \in Bot$› **and**
deriv: ‹*chain* (▷) *Ns*› **and**
fair: ‹*fair Ns*› **and**
unsat: ‹*lhd Ns* $\models \{B\}$›
shows ‹$\exists B' \in Bot.\ B' \in$ *Liminf-llist Ns*›
⟨*proof*⟩

end

locale *dynamically-complete-calculus* = *calculus* +
assumes
dynamically-complete: $B \in Bot \Longrightarrow$ *chain* (▷) *Ns* $\Longrightarrow$ *fair Ns* $\Longrightarrow$ *lhd Ns* $\models \{B\} \Longrightarrow$
$\exists i \in \{i.\ enat\ i < llength\ Ns\}.\ \exists B' \in Bot.\ B' \in lnth\ Ns\ i$
begin

sublocale *statically-complete-calculus*
⟨*proof*⟩

end

sublocale *statically-complete-calculus* ⊆ *dynamically-complete-calculus*
⟨*proof*⟩

end

2 Calculi Based on the Intersection of Redundancy Criteria

In this section, section 2.3 of the report is covered, on calculi equipped with a family of redundancy criteria.

theory *Intersection-Calculus*
imports
Calculus
Ordered-Resolution-Prover.Lazy-List-Liminf
Ordered-Resolution-Prover.Lazy-List-Chain
begin

2.1 Calculi with a Family of Redundancy Criteria

locale *intersection-calculus* =

inference-system Inf + *consequence-relation-family Bot Q entails-q*
for
Bot :: *'f set* **and**
Inf :: ‹*'f inference set*› **and**
Q :: *'q set* **and**
entails-q :: $'q \Rightarrow$ *'f set* $\Rightarrow$ *'f set* $\Rightarrow$ *bool*
\+ **fixes**
Red-I-q :: $'q \Rightarrow$ *'f set* $\Rightarrow$ *'f inference set* **and**
Red-F-q :: $'q \Rightarrow$ *'f set* $\Rightarrow$ *'f set*
assumes
Q-nonempty: $Q \neq \{\}$ **and**
all-red-crit: $\forall\, q \in Q$. *calculus Bot Inf* (*entails-q q*) (*Red-I-q q*) (*Red-F-q q*)
begin

definition *Red-I* :: *'f set* $\Rightarrow$ *'f inference set* **where**
Red-I N = ($\bigcap q \in Q$. *Red-I-q q N*)

definition *Red-F* :: *'f set* $\Rightarrow$ *'f set* **where**
Red-F N = ($\bigcap q \in Q$. *Red-F-q q N*)

sublocale *calculus Bot Inf entails Red-I Red-F*
⟨*proof*⟩

lemma *sat-int-to-sat-q*: *calculus.saturated Inf Red-I N* $\longleftrightarrow$
($\forall\, qi \in Q$. *calculus.saturated Inf* (*Red-I-q qi*) *N*) **for** *N*
⟨*proof*⟩

lemma *stat-ref-comp-from-bot-in-sat*:
($\forall\, N$. *calculus.saturated Inf Red-I N* $\wedge$ ($\forall\, B \in Bot.\ B \notin N$) $\longrightarrow$
($\exists\, B \in Bot.\ \exists\, qi \in Q.\ \neg$ *entails-q qi N* $\{B\}$)) $\Longrightarrow$
statically-complete-calculus Bot Inf entails Red-I Red-F
⟨*proof*⟩

end

end

3 Variations on a Theme

In this section, section 2.4 of the report is covered, demonstrating that various notions of redundancy are equivalent.

theory *Calculus-Variations*
imports *Calculus*
begin

locale *reduced-calculus* = *calculus Bot Inf entails Red-I Red-F*
for
Bot :: *'f set* **and**
Inf :: ‹*'f inference set*› **and**
entails :: *'f set* $\Rightarrow$ *'f set* $\Rightarrow$ *bool* (**infix** ‹$\models$› *50*) **and**
Red-I :: *'f set* $\Rightarrow$ *'f inference set* **and**

Red-F :: *'f set* ⇒ *'f set*
+ **assumes**
inf-in-red-inf: *Inf-between UNIV* (*Red-F N*) ⊆ *Red-I N*
begin

lemma *sat-eq-reduc-sat*: *saturated N* ⟷ *reduc-saturated N*
⟨*proof*⟩

end

locale *reducedly-statically-complete-calculus* = *calculus* +
assumes *reducedly-statically-complete*:
$B \in Bot \Longrightarrow$ *reduc-saturated* $N \Longrightarrow N \models \{B\} \Longrightarrow \exists B' \in Bot.\ B' \in N$

locale *reducedly-statically-complete-reduced-calculus* = *reduced-calculus* +
assumes *reducedly-statically-complete*:
$B \in Bot \Longrightarrow$ *reduc-saturated* $N \Longrightarrow N \models \{B\} \Longrightarrow \exists B' \in Bot.\ B' \in N$
begin

sublocale *reducedly-statically-complete-calculus*
⟨*proof*⟩

sublocale *statically-complete-calculus*
⟨*proof*⟩

end

context *reduced-calculus*
begin

lemma *stat-ref-comp-imp-red-stat-ref-comp*:
statically-complete-calculus Bot Inf entails Red-I Red-F ⟹
reducedly-statically-complete-calculus Bot Inf entails Red-I Red-F
⟨*proof*⟩

end

context *calculus*
begin

definition *Red-Red-I* :: *'f set* ⇒ *'f inference set* **where**
Red-Red-I N = *Red-I N* ∪ *Inf-between UNIV* (*Red-F N*)

lemma *reduced-calc-is-calc*: *calculus Bot Inf entails Red-Red-I Red-F*
⟨*proof*⟩

lemma *inf-subs-reduced-red-inf*: *Inf-between UNIV* (*Red-F N*) ⊆ *Red-Red-I N*
⟨*proof*⟩

The following is a lemma and not a sublocale as was previously used in similar cases. Here, a sublocale cannot be used because it would create an infinitely descending chain of sublocales.

lemma *reduc-calc*: *reduced-calculus Bot Inf entails Red-Red-I Red-F*

⟨*proof*⟩

interpretation *reduc-calc*: *reduced-calculus Bot Inf entails Red-Red-I Red-F*
⟨*proof*⟩

lemma *sat-imp-red-calc-sat*: *saturated N* $\Longrightarrow$ *reduc-calc.saturated N*
⟨*proof*⟩

lemma *red-sat-eq-red-calc-sat*: *reduc-saturated N* $\longleftrightarrow$ *reduc-calc.saturated N*
⟨*proof*⟩

lemma *red-sat-eq-sat*: *reduc-saturated N* $\longleftrightarrow$ *saturated* (*N* − *Red-F N*)
⟨*proof*⟩

theorem *stat-is-stat-red*: *statically-complete-calculus Bot Inf entails Red-I Red-F* $\longleftrightarrow$
statically-complete-calculus Bot Inf entails Red-Red-I Red-F
⟨*proof*⟩

theorem *red-stat-red-is-stat-red*:
reducedly-statically-complete-calculus Bot Inf entails Red-Red-I Red-F $\longleftrightarrow$
statically-complete-calculus Bot Inf entails Red-Red-I Red-F
⟨*proof*⟩

theorem *red-stat-is-stat-red*:
reducedly-statically-complete-calculus Bot Inf entails Red-I Red-F $\longleftrightarrow$
statically-complete-calculus Bot Inf entails Red-Red-I Red-F
⟨*proof*⟩

lemma *sup-red-f-in-red-liminf*:
chain derive Ns $\Longrightarrow$ *Sup-llist* (*lmap Red-F Ns*) $\subseteq$ *Red-F* (*Liminf-llist Ns*)
⟨*proof*⟩

lemma *sup-red-inf-in-red-liminf*:
chain derive Ns $\Longrightarrow$ *Sup-llist* (*lmap Red-I Ns*) $\subseteq$ *Red-I* (*Liminf-llist Ns*)
⟨*proof*⟩

definition *reduc-fair* :: *'f set llist* ⇒ *bool* **where**
reduc-fair Ns $\longleftrightarrow$
Inf-from (*Liminf-llist Ns* − *Sup-llist* (*lmap Red-F Ns*)) $\subseteq$ *Sup-llist* (*lmap Red-I Ns*)

lemma *reduc-fair-imp-Liminf-reduc-sat*:
chain derive Ns $\Longrightarrow$ *reduc-fair Ns* $\Longrightarrow$ *reduc-saturated* (*Liminf-llist Ns*)
⟨*proof*⟩

end

locale *reducedly-dynamically-complete-calculus* = *calculus* +
assumes

reducedly-dynamically-complete: $B \in Bot \Longrightarrow chain\ derive\ Ns \Longrightarrow reduc\text{-}fair\ Ns \Longrightarrow$
$lhd\ Ns \models \{B\} \Longrightarrow \exists\, i \in \{i.\ enat\ i < llength\ Ns\}.\ \exists\ B' \in Bot.\ B' \in lnth\ Ns\ i$
begin

sublocale *reducedly-statically-complete-calculus*
⟨*proof*⟩

end

sublocale *reducedly-statically-complete-calculus* ⊆ *reducedly-dynamically-complete-calculus*
⟨*proof*⟩

context *calculus*
begin

lemma *dyn-equiv-stat*: *dynamically-complete-calculus Bot Inf entails Red-I Red-F* =
statically-complete-calculus Bot Inf entails Red-I Red-F
⟨*proof*⟩

lemma *red-dyn-equiv-red-stat*:
reducedly-dynamically-complete-calculus Bot Inf entails Red-I Red-F =
reducedly-statically-complete-calculus Bot Inf entails Red-I Red-F
⟨*proof*⟩

interpretation *reduc-calc*: *reduced-calculus Bot Inf entails Red-Red-I Red-F*
⟨*proof*⟩

theorem *dyn-ref-eq-dyn-ref-red*:
dynamically-complete-calculus Bot Inf entails Red-I Red-F ⟷
dynamically-complete-calculus Bot Inf entails Red-Red-I Red-F
⟨*proof*⟩

theorem *red-dyn-ref-red-eq-dyn-ref-red*:
reducedly-dynamically-complete-calculus Bot Inf entails Red-Red-I Red-F ⟷
dynamically-complete-calculus Bot Inf entails Red-Red-I Red-F
⟨*proof*⟩

theorem *red-dyn-ref-eq-dyn-ref-red*:
reducedly-dynamically-complete-calculus Bot Inf entails Red-I Red-F ⟷
dynamically-complete-calculus Bot Inf entails Red-Red-I Red-F
⟨*proof*⟩

end

end

4 Lifting to Non-ground Calculi

The section 3.1 to 3.3 of the report are covered by the current section. Various forms of lifting are proven correct. These allow to obtain the dynamic refutational completeness of a non-ground calculus from the static refutational completeness of its ground counterpart.

theory *Lifting-to-Non-Ground-Calculi*
imports
Intersection-Calculus
Calculus-Variations
begin

4.1 Standard Lifting

locale *standard-lifting* = *inference-system Inf-F* +
ground: *calculus Bot-G Inf-G entails-G Red-I-G Red-F-G*
for
Inf-F :: ‹′*f inference set*› **and**
Bot-G :: ‹′*g set*› **and**
Inf-G :: ‹′*g inference set*› **and**
entails-G :: ‹′*g set* ⇒ ′*g set* ⇒ *bool*› (**infix** ‹$\models$*G*› *50*) **and**
Red-I-G :: ‹′*g set* ⇒ ′*g inference set*› **and**
Red-F-G :: ‹′*g set* ⇒ ′*g set*›
+ **fixes**
Bot-F :: ‹′*f set*› **and**
$\mathcal{G}$-*F* :: ‹′*f* ⇒ ′*g set*› **and**
$\mathcal{G}$-*I* :: ‹′*f inference* ⇒ ′*g inference set option*›
assumes
Bot-F-not-empty: *Bot-F* ≠ {} **and**
Bot-map-not-empty: ‹*B* ∈ *Bot-F* ⟹ $\mathcal{G}$-*F B* ≠ {}› **and**
Bot-map: ‹*B* ∈ *Bot-F* ⟹ $\mathcal{G}$-*F B* ⊆ *Bot-G*› **and**
Bot-cond: ‹$\mathcal{G}$-*F C* ∩ *Bot-G* ≠ {} ⟶ *C* ∈ *Bot-F*› **and**
inf-map: ‹ι ∈ *Inf-F* ⟹ $\mathcal{G}$-*I* ι ≠ *None* ⟹ *the* ($\mathcal{G}$-*I* ι) ⊆ *Red-I-G* ($\mathcal{G}$-*F* (*concl-of* ι))›
begin

abbreviation $\mathcal{G}$-*Fset* :: ‹′*f set* ⇒ ′*g set*› **where**
‹$\mathcal{G}$-*Fset N* ≡ ⋃ ($\mathcal{G}$-*F* ‘ *N*)›

lemma $\mathcal{G}$-*subset*: ‹*N1* ⊆ *N2* ⟹ $\mathcal{G}$-*Fset N1* ⊆ $\mathcal{G}$-*Fset N2*› ⟨*proof*⟩

abbreviation *entails*-$\mathcal{G}$ :: ‹′*f set* ⇒ ′*f set* ⇒ *bool*› (**infix** ‹$\models\mathcal{G}$› *50*) **where**
‹*N1* $\models\mathcal{G}$ *N2* ≡ $\mathcal{G}$-*Fset N1* $\models$*G* $\mathcal{G}$-*Fset N2*›

lemma *subs-Bot-G-entails*:
assumes
not-empty: ‹*sB* ≠ {}› **and**
in-bot: ‹*sB* ⊆ *Bot-G*›
shows ‹*sB* $\models$*G N*›
⟨*proof*⟩

sublocale *consequence-relation Bot-F entails*-$\mathcal{G}$
⟨*proof*⟩

definition *Red-I*-$\mathcal{G}$ :: ′*f set* ⇒ ′*f inference set* **where**
‹*Red-I*-$\mathcal{G}$ *N* = {ι ∈ *Inf-F*. ($\mathcal{G}$-*I* ι ≠ *None* ∧ *the* ($\mathcal{G}$-*I* ι) ⊆ *Red-I-G* ($\mathcal{G}$-*Fset N*))
∨ ($\mathcal{G}$-*I* ι = *None* ∧ $\mathcal{G}$-*F* (*concl-of* ι) ⊆ $\mathcal{G}$-*Fset N* ∪ *Red-F-G* ($\mathcal{G}$-*Fset N*))}›

definition *Red-F*-$\mathcal{G}$ :: ′*f set* ⇒ ′*f set* **where**
‹*Red-F*-$\mathcal{G}$ *N* = {*C*. ∀ *D* ∈ $\mathcal{G}$-*F C*. *D* ∈ *Red-F-G* ($\mathcal{G}$-*Fset N*)}›
end

4.2 Strong Standard Lifting

locale *strong-standard-lifting* = *inference-system Inf-F* +
  ground: *calculus Bot-G Inf-G entails-G Red-I-G Red-F-G*
  for
    Inf-F :: ‹′*f inference set*› **and**
    Bot-G :: ‹′*g set*› **and**
    Inf-G :: ‹′*g inference set*› **and**
    entails-G :: ‹′*g set* ⇒ ′*g set* ⇒ *bool*› (**infix** ‹⊨*G*› *50*) **and**
    Red-I-G :: ‹′*g set* ⇒ ′*g inference set*› **and**
    Red-F-G :: ‹′*g set* ⇒ ′*g set*›
  + **fixes**
    Bot-F :: ‹′*f set*› **and**
    𝒢-F :: ‹′*f* ⇒ ′*g set*› **and**
    𝒢-I :: ‹′*f inference* ⇒ ′*g inference set option*›
  assumes
    Bot-F-not-empty: *Bot-F* ≠ {} **and**
    Bot-map-not-empty: ‹*B* ∈ *Bot-F* ⟹ *𝒢-F B* ≠ {}› **and**
    Bot-map: ‹*B* ∈ *Bot-F* ⟹ *𝒢-F B* ⊆ *Bot-G*› **and**
    Bot-cond: ‹*𝒢-F C* ∩ *Bot-G* ≠ {} ⟶ *C* ∈ *Bot-F*› **and**
    strong-inf-map: ‹*ι* ∈ *Inf-F* ⟹ *𝒢-I ι* ≠ *None* ⟹ *concl-of* ‘ (*the* (*𝒢-I ι*)) ⊆ (*𝒢-F* (*concl-of ι*))› **and**
    inf-map-in-Inf: ‹*ι* ∈ *Inf-F* ⟹ *𝒢-I ι* ≠ *None* ⟹ *the* (*𝒢-I ι*) ⊆ *Inf-G*›
begin

sublocale *standard-lifting Inf-F Bot-G Inf-G* (⊨*G*) *Red-I-G Red-F-G Bot-F 𝒢-F 𝒢-I*
⟨*proof*⟩

end

4.3 Lifting with a Family of Tiebreaker Orderings

locale *tiebreaker-lifting* =
  empty-ord?: *standard-lifting Inf-F Bot-G Inf-G entails-G Red-I-G Red-F-G Bot-F 𝒢-F 𝒢-I*
  for
    Bot-F :: ‹′*f set*› **and**
    Inf-F :: ‹′*f inference set*› **and**
    Bot-G :: ‹′*g set*› **and**
    entails-G :: ‹′*g set* ⇒ ′*g set* ⇒ *bool*› (**infix** ‹⊨*G*› *50*) **and**
    Inf-G :: ‹′*g inference set*› **and**
    Red-I-G :: ‹′*g set* ⇒ ′*g inference set*› **and**
    Red-F-G :: ‹′*g set* ⇒ ′*g set*› **and**
    𝒢-F :: ′*f* ⇒ ′*g set* **and**
    𝒢-I :: ′*f inference* ⇒ ′*g inference set option*
  + **fixes**
    Prec-F-g :: ‹′*g* ⇒ ′*f* ⇒ ′*f* ⇒ *bool*›
  assumes
    all-wf: *wfp* (*Prec-F-g g*) *transp* (*Prec-F-g g*)
begin

definition *Red-F-𝒢* :: ′*f set* ⇒ ′*f set* **where**
  ‹*Red-F-𝒢 N* = {*C*. ∀ *D* ∈ *𝒢-F C*. *D* ∈ *Red-F-G* (*𝒢-Fset N*) ∨ (∃ *E* ∈ *N*. *Prec-F-g D E C* ∧ *D* ∈ *𝒢-F E*)}›

lemma *Prec-trans*:
  assumes
    ‹*Prec-F-g D A B*› **and**

‹*Prec-F-g* $D\ B\ C$›
shows
‹*Prec-F-g* $D\ A\ C$›
⟨*proof*⟩

lemma *prop-nested-in-set*: $D \in P\ C \Longrightarrow C \in \{C.\ \forall D \in P\ C.\ A\ D \vee B\ C\ D\} \Longrightarrow A\ D \vee B\ C\ D$
⟨*proof*⟩

lemma *Red-F-𝒢-equiv-def*:
‹$\mathit{Red\text{-}F\text{-}\mathcal{G}}\ N = \{C.\ \forall Di \in \mathcal{G}\text{-}F\ C.\ Di \in \mathit{Red\text{-}F\text{-}G}\ (\mathcal{G}\text{-}\mathit{Fset}\ N) \vee$
$(\exists E \in (N - \mathit{Red\text{-}F\text{-}\mathcal{G}}\ N).\ \mathit{Prec\text{-}F\text{-}g}\ Di\ E\ C \wedge Di \in \mathcal{G}\text{-}F\ E)\}$›
⟨*proof*⟩

lemma *not-red-map-in-map-not-red*: ‹$\mathcal{G}\text{-}\mathit{Fset}\ N - \mathit{Red\text{-}F\text{-}G}\ (\mathcal{G}\text{-}\mathit{Fset}\ N) \subseteq \mathcal{G}\text{-}\mathit{Fset}\ (N - \mathit{Red\text{-}F\text{-}\mathcal{G}}\ N)$›
⟨*proof*⟩

lemma *Red-F-Bot-F*: ‹$B \in \mathit{Bot\text{-}F} \Longrightarrow N \models\mathcal{G}\ \{B\} \Longrightarrow N - \mathit{Red\text{-}F\text{-}\mathcal{G}}\ N \models\mathcal{G}\ \{B\}$›
⟨*proof*⟩

lemma *Red-F-of-subset-F*: ‹$N \subseteq N' \Longrightarrow \mathit{Red\text{-}F\text{-}\mathcal{G}}\ N \subseteq \mathit{Red\text{-}F\text{-}\mathcal{G}}\ N'$›
⟨*proof*⟩

lemma *Red-I-of-subset-F*: ‹$N \subseteq N' \Longrightarrow \mathit{Red\text{-}I\text{-}\mathcal{G}}\ N \subseteq \mathit{Red\text{-}I\text{-}\mathcal{G}}\ N'$›
⟨*proof*⟩

lemma *Red-F-of-Red-F-subset-F*: ‹$N' \subseteq \mathit{Red\text{-}F\text{-}\mathcal{G}}\ N \Longrightarrow \mathit{Red\text{-}F\text{-}\mathcal{G}}\ N \subseteq \mathit{Red\text{-}F\text{-}\mathcal{G}}\ (N - N')$›
⟨*proof*⟩

lemma *Red-I-of-Red-F-subset-F*: ‹$N' \subseteq \mathit{Red\text{-}F\text{-}\mathcal{G}}\ N \Longrightarrow \mathit{Red\text{-}I\text{-}\mathcal{G}}\ N \subseteq \mathit{Red\text{-}I\text{-}\mathcal{G}}\ (N - N')$ ›
⟨*proof*⟩

lemma *Red-I-of-Inf-to-N-F*:
assumes
i-in: ‹$\iota \in \mathit{Inf\text{-}F}$› **and**
concl-i-in: ‹$\mathit{concl\text{-}of}\ \iota \in N$›
shows
‹$\iota \in \mathit{Red\text{-}I\text{-}\mathcal{G}}\ N$ ›
⟨*proof*⟩

sublocale *calculus Bot-F Inf-F entails-𝒢 Red-I-𝒢 Red-F-𝒢*
⟨*proof*⟩

end

lemma *standard-empty-tiebreaker-equiv*: *standard-lifting Inf-F Bot-G Inf-G entails-G Red-I-G Red-F-G Bot-F 𝒢-F 𝒢-I* = *tiebreaker-lifting Bot-F Inf-F Bot-G entails-G Inf-G Red-I-G*

Red-F-G $\mathcal{G}$-F $\mathcal{G}$-I (λg *C C'. False*)
⟨*proof*⟩

context *standard-lifting*
begin

interpretation *empt-ord*: *tiebreaker-lifting Bot-F Inf-F Bot-G entails-G Inf-G Red-I-G Red-F-G $\mathcal{G}$-F $\mathcal{G}$-I λg C C'. False*
⟨*proof*⟩

lemma *red-f-equiv*: *empt-ord.Red-F-$\mathcal{G}$* = *Red-F-$\mathcal{G}$*
⟨*proof*⟩

sublocale *calc?*: *calculus Bot-F Inf-F entails-$\mathcal{G}$ Red-I-$\mathcal{G}$ Red-F-$\mathcal{G}$*
⟨*proof*⟩

lemma *grounded-inf-in-ground-inf*: $\iota \in$ *Inf-F* $\Longrightarrow$ *$\mathcal{G}$-I* $\iota \neq$ *None* $\Longrightarrow$ *the* (*$\mathcal{G}$-I* ι) $\subseteq$ *Inf-G*
⟨*proof*⟩

abbreviation *ground-Inf-overapproximated* :: *'f set* $\Rightarrow$ *bool* **where**
ground-Inf-overapproximated N $\equiv$ *ground.Inf-from* (*$\mathcal{G}$-Fset N*)
$\subseteq$ {ι. $\exists\,\iota' \in$ *Inf-from N*. *$\mathcal{G}$-I* $\iota' \neq$ *None* $\wedge$ $\iota \in$ *the* (*$\mathcal{G}$-I* ι')} $\cup$ *Red-I-G* (*$\mathcal{G}$-Fset N*)

lemma *sat-inf-imp-ground-red*:
assumes
saturated N **and**
$\iota' \in$ *Inf-from N* **and**
$\mathcal{G}$-I $\iota' \neq$ *None* $\wedge$ $\iota \in$ *the* (*$\mathcal{G}$-I* ι')
shows $\iota \in$ *Red-I-G* (*$\mathcal{G}$-Fset N*)
⟨*proof*⟩

lemma *sat-imp-ground-sat*:
saturated N $\Longrightarrow$ *ground-Inf-overapproximated N* $\Longrightarrow$ *ground.saturated* (*$\mathcal{G}$-Fset N*)
⟨*proof*⟩

theorem *stat-ref-comp-to-non-ground*:
assumes
stat-ref-G: *statically-complete-calculus Bot-G Inf-G entails-G Red-I-G Red-F-G* **and**
sat-n-imp: $\bigwedge N$. *saturated N* $\Longrightarrow$ *ground-Inf-overapproximated N*
shows
statically-complete-calculus Bot-F Inf-F entails-$\mathcal{G}$ Red-I-$\mathcal{G}$ Red-F-$\mathcal{G}$
⟨*proof*⟩

end

context *tiebreaker-lifting*
begin

lemma *saturated-empty-order-equiv-saturated*:
saturated N = *calc.saturated N*

⟨*proof*⟩

lemma *static-empty-order-equiv-static*:
statically-complete-calculus Bot-F Inf-F entails-$\mathcal{G}$ Red-I-$\mathcal{G}$ Red-F-$\mathcal{G}$ =
statically-complete-calculus Bot-F Inf-F entails-$\mathcal{G}$ Red-I-$\mathcal{G}$ empty-ord.Red-F-$\mathcal{G}$
⟨*proof*⟩

theorem *static-to-dynamic*:
statically-complete-calculus Bot-F Inf-F entails-$\mathcal{G}$ Red-I-$\mathcal{G}$ empty-ord.Red-F-$\mathcal{G}$ =
dynamically-complete-calculus Bot-F Inf-F entails-$\mathcal{G}$ Red-I-$\mathcal{G}$ Red-F-$\mathcal{G}$
⟨*proof*⟩

end

4.4 Lifting with a Family of Redundancy Criteria

locale *lifting-intersection* = *inference-system Inf-F* +
ground: *inference-system-family Q Inf-G-q* +
ground: *consequence-relation-family Bot-G Q entails-q*
for
Inf-F :: *'f inference set* **and**
Bot-G :: *'g set* **and**
Q :: *'q set* **and**
Inf-G-q :: ‹*'q* ⇒ *'g inference set*› **and**
entails-q :: *'q* ⇒ *'g set* ⇒ *'g set* ⇒ *bool* **and**
Red-I-q :: *'q* ⇒ *'g set* ⇒ *'g inference set* **and**
Red-F-q :: *'q* ⇒ *'g set* ⇒ *'g set*
\+ **fixes**
Bot-F :: *'f set* **and**
$\mathcal{G}$-F-q :: *'q* ⇒ *'f* ⇒ *'g set* **and**
$\mathcal{G}$-I-q :: *'q* ⇒ *'f inference* ⇒ *'g inference set option* **and**
Prec-F-g :: *'g* ⇒ *'f* ⇒ *'f* ⇒ *bool*
assumes
standard-lifting-family:
∀ *q* ∈ *Q*. *tiebreaker-lifting Bot-F Inf-F Bot-G* (*entails-q q*) (*Inf-G-q q*) (*Red-I-q q*)
(*Red-F-q q*) (*$\mathcal{G}$-F-q q*) (*$\mathcal{G}$-I-q q*) *Prec-F-g*
begin

abbreviation *$\mathcal{G}$-Fset-q* :: *'q* ⇒ *'f set* ⇒ *'g set* **where**
$\mathcal{G}$-Fset-q q N ≡ ⋃ (*$\mathcal{G}$-F-q q* ‘ *N*)

definition *Red-I-$\mathcal{G}$-q* :: *'q* ⇒ *'f set* ⇒ *'f inference set* **where**
Red-I-$\mathcal{G}$-q q N = {*ι* ∈ *Inf-F*. (*$\mathcal{G}$-I-q q ι* ≠ *None* ∧ *the* (*$\mathcal{G}$-I-q q ι*) ⊆ *Red-I-q q* (*$\mathcal{G}$-Fset-q q N*))
∨ (*$\mathcal{G}$-I-q q ι* = *None* ∧ *$\mathcal{G}$-F-q q* (*concl-of ι*) ⊆ (*$\mathcal{G}$-Fset-q q N* ∪ *Red-F-q q* (*$\mathcal{G}$-Fset-q q N*)))}

definition *Red-F-$\mathcal{G}$-empty-q* :: *'q* ⇒ *'f set* ⇒ *'f set* **where**
Red-F-$\mathcal{G}$-empty-q q N = {*C*. ∀ *D* ∈ *$\mathcal{G}$-F-q q C*. *D* ∈ *Red-F-q q* (*$\mathcal{G}$-Fset-q q N*)}

definition *Red-F-$\mathcal{G}$-q* :: *'q* ⇒ *'f set* ⇒ *'f set* **where**
Red-F-$\mathcal{G}$-q q N =
{*C*. ∀ *D* ∈ *$\mathcal{G}$-F-q q C*. *D* ∈ *Red-F-q q* (*$\mathcal{G}$-Fset-q q N*) ∨ (∃ *E* ∈ *N*. *Prec-F-g D E C* ∧ *D* ∈ *$\mathcal{G}$-F-q q*
E)}

abbreviation *entails-$\mathcal{G}$-q* :: *'q* ⇒ *'f set* ⇒ *'f set* ⇒ *bool* **where**

entails-𝒢-q q N1 N2 ≡ *entails-q q* (*𝒢-Fset-q q N1*) (*𝒢-Fset-q q N2*)

lemma *red-crit-lifting-family*:
assumes *q-in*: $q \in Q$
shows *calculus Bot-F Inf-F* (*entails-𝒢-q q*) (*Red-I-𝒢-q q*) (*Red-F-𝒢-q q*)
⟨*proof*⟩

lemma *red-crit-lifting-family-empty-ord*:
assumes *q-in*: $q \in Q$
shows *calculus Bot-F Inf-F* (*entails-𝒢-q q*) (*Red-I-𝒢-q q*) (*Red-F-𝒢-empty-q q*)
⟨*proof*⟩

sublocale *consequence-relation-family Bot-F Q entails-𝒢-q*
⟨*proof*⟩

sublocale *intersection-calculus Bot-F Inf-F Q entails-𝒢-q Red-I-𝒢-q Red-F-𝒢-q*
⟨*proof*⟩

abbreviation *entails-𝒢* :: *'f set* ⇒ *'f set* ⇒ *bool* (**infix** ‹⊨∩𝒢› *50*) **where**
(⊨∩𝒢) ≡ *entails*

abbreviation *Red-I-𝒢* :: *'f set* ⇒ *'f inference set* **where**
Red-I-𝒢 ≡ *Red-I*

abbreviation *Red-F-𝒢* :: *'f set* ⇒ *'f set* **where**
Red-F-𝒢 ≡ *Red-F*

lemmas *entails-𝒢-def* = *entails-def*
lemmas *Red-I-𝒢-def* = *Red-I-def*
lemmas *Red-F-𝒢-def* = *Red-F-def*

sublocale *empty-ord*: *intersection-calculus Bot-F Inf-F Q entails-𝒢-q Red-I-𝒢-q Red-F-𝒢-empty-q*
⟨*proof*⟩

abbreviation *Red-F-𝒢-empty* :: *'f set* ⇒ *'f set* **where**
Red-F-𝒢-empty ≡ *empty-ord.Red-F*

lemmas *Red-F-𝒢-empty-def* = *empty-ord.Red-F-def*

lemma *sat-inf-imp-ground-red-fam-inter*:
assumes
sat-n: *saturated N* **and**
i'-in: $\iota' \in$ *Inf-from N* **and**
q-in: $q \in Q$ **and**
grounding: *𝒢-I-q q* $\iota' \neq$ *None* ∧ $\iota \in$ *the* (*𝒢-I-q q* ι')
shows $\iota \in$ *Red-I-q q* (*𝒢-Fset-q q N*)
⟨*proof*⟩

abbreviation *ground-Inf-overapproximated* :: *'q* ⇒ *'f set* ⇒ *bool* **where**
ground-Inf-overapproximated q N ≡
ground.Inf-from-q q (*𝒢-Fset-q q N*)
⊆ {ι. ∃ι'∈ *Inf-from N*. *𝒢-I-q q* $\iota' \neq$ *None* ∧ $\iota \in$ *the* (*𝒢-I-q q* ι')} ∪ *Red-I-q q* (*𝒢-Fset-q q N*)

abbreviation *ground-saturated* :: *'q* ⇒ *'f set* ⇒ *bool* **where**
ground-saturated q N ≡ *ground.Inf-from-q q* (*𝒢-Fset-q q N*) ⊆ *Red-I-q q* (*𝒢-Fset-q q N*)

lemma *sat-imp-ground-sat-fam-inter*:
saturated N $\Longrightarrow$ *q* $\in$ *Q* $\Longrightarrow$ *ground-Inf-overapproximated q N* $\Longrightarrow$ *ground-saturated q N*
⟨*proof*⟩

theorem *stat-ref-comp-to-non-ground-fam-inter*:
assumes
stat-ref-G:
$\forall$ *q* $\in$ *Q*. *statically-complete-calculus Bot-G* (*Inf-G-q q*) (*entails-q q*) (*Red-I-q q*) (*Red-F-q q*) **and**
sat-n-imp: $\bigwedge$*N*. *saturated N* $\Longrightarrow$ $\exists$ *q* $\in$ *Q*. *ground-Inf-overapproximated q N*
shows
statically-complete-calculus Bot-F Inf-F entails-$\mathcal{G}$ *Red-I-*$\mathcal{G}$ *Red-F-*$\mathcal{G}$*-empty*
⟨*proof*⟩

lemma *sat-eq-sat-empty-order*: *saturated N* = *empty-ord.saturated N*
⟨*proof*⟩

lemma *static-empty-ord-inter-equiv-static-inter*:
statically-complete-calculus Bot-F Inf-F entails Red-I Red-F =
statically-complete-calculus Bot-F Inf-F entails Red-I Red-F-$\mathcal{G}$*-empty*
⟨*proof*⟩

theorem *stat-eq-dyn-ref-comp-fam-inter*: *statically-complete-calculus Bot-F Inf-F entails Red-I Red-F-*$\mathcal{G}$*-empty* =
dynamically-complete-calculus Bot-F Inf-F entails Red-I Red-F
⟨*proof*⟩

end

end

5 Labeled Lifting to Non-Ground Calculi

This section formalizes the extension of the lifting results to labeled calculi. This corresponds to section 3.4 of the report.

theory *Labeled-Lifting-to-Non-Ground-Calculi*
imports *Lifting-to-Non-Ground-Calculi*
begin

5.1 Labeled Lifting with a Family of Tiebreaker Orderings

locale *labeled-tiebreaker-lifting* = *no-labels*: *tiebreaker-lifting Bot-F Inf-F Bot-G entails-G Inf-G Red-I-G Red-F-G* $\mathcal{G}$*-F* $\mathcal{G}$*-I Prec-F*
for
Bot-F :: *'f set* **and**
Inf-F :: *'f inference set* **and**
Bot-G :: *'g set* **and**
entails-G :: *'g set* $\Rightarrow$ *'g set* $\Rightarrow$ *bool* (**infix** ‹$\models$*G*› *50*) **and**
Inf-G :: *'g inference set* **and**

Red-I-G :: *'g set* ⇒ *'g inference set* **and**
Red-F-G :: *'g set* ⇒ *'g set* **and**
$\mathcal{G}$*-F* :: *'f* ⇒ *'g set* **and**
$\mathcal{G}$*-I* :: *'f inference* ⇒ *'g inference set option* **and**
Prec-F :: *'g* ⇒ *'f* ⇒ *'f* ⇒ *bool* (**infix** ‹⊏› *50*)
+ **fixes**
Inf-FL :: ‹(*'f* × *'l*) *inference set*›
assumes
Inf-F-to-Inf-FL: ‹ι_F ∈ *Inf-F* ⟹ *length* (*Ll* :: *'l list*) = *length* (*prems-of* ι_F) ⟹
∃ *L0*. *Infer* (*zip* (*prems-of* ι_F) *Ll*) (*concl-of* ι_F, *L0*) ∈ *Inf-FL*› **and**
Inf-FL-to-Inf-F: ‹ι_{FL} ∈ *Inf-FL* ⟹ *Infer* (*map fst* (*prems-of* ι_{FL})) (*fst* (*concl-of* ι_{FL})) ∈ *Inf-F*›
begin

definition *to-F* :: ‹(*'f* × *'l*) *inference* ⇒ *'f inference*› **where**
‹*to-F* ι_{FL} = *Infer* (*map fst* (*prems-of* ι_{FL})) (*fst* (*concl-of* ι_{FL}))›

abbreviation *Bot-FL* :: ‹(*'f* × *'l*) *set*› **where**
‹*Bot-FL* ≡ *Bot-F* × *UNIV*›

abbreviation $\mathcal{G}$*-F-L* :: ‹(*'f* × *'l*) ⇒ *'g set*› **where**
‹$\mathcal{G}$*-F-L CL* ≡ $\mathcal{G}$*-F* (*fst CL*)›

abbreviation $\mathcal{G}$*-I-L* :: ‹(*'f* × *'l*) *inference* ⇒ *'g inference set option*› **where**
‹$\mathcal{G}$*-I-L* ι_{FL} ≡ $\mathcal{G}$*-I* (*to-F* ι_{FL})›

sublocale *standard-lifting Inf-FL Bot-G Inf-G* (⊨*G*) *Red-I-G Red-F-G Bot-FL* $\mathcal{G}$*-F-L* $\mathcal{G}$*-I-L*
⟨*proof*⟩

notation *entails-*$\mathcal{G}$ (**infix** ‹⊨$\mathcal{G}$*L*› *50*)

lemma *labeled-entailment-lifting*: *NL1* ⊨$\mathcal{G}$*L NL2* ⟷ *fst* ‘ *NL1* ⊨$\mathcal{G}$ *fst* ‘ *NL2*
⟨*proof*⟩

lemma *red-inf-impl*: ι ∈ *Red-I-*$\mathcal{G}$ *NL* ⟹ *to-F* ι ∈ *no-labels.Red-I-*$\mathcal{G}$ (*fst* ‘ *NL*)
⟨*proof*⟩

lemma *labeled-saturation-lifting*: *saturated NL* ⟹ *no-labels.saturated* (*fst* ‘ *NL*)
⟨*proof*⟩

lemma *stat-ref-comp-to-labeled-sta-ref-comp*:
assumes *static*:
statically-complete-calculus Bot-F Inf-F (⊨$\mathcal{G}$) *no-labels.Red-I-*$\mathcal{G}$ *no-labels.Red-F-*$\mathcal{G}$
shows *statically-complete-calculus Bot-FL Inf-FL* (⊨$\mathcal{G}$*L*) *Red-I-*$\mathcal{G}$ *Red-F-*$\mathcal{G}$
⟨*proof*⟩

end

5.2 Labeled Lifting with a Family of Redundancy Criteria

locale *labeled-lifting-intersection* = *no-labels*: *lifting-intersection Inf-F*

Bot-G Q Inf-G-q entails-q Red-I-q Red-F-q Bot-F $\mathcal{G}$*-F-q* $\mathcal{G}$*-I-q* λg *Cl Cl'. False*
for
Bot-F :: *'f set* **and**
Inf-F :: *'f inference set* **and**
Bot-G :: *'g set* **and**
Q :: *'q set* **and**
entails-q :: *'q* ⇒ *'g set* ⇒ *'g set* ⇒ *bool* **and**
Inf-G-q :: *'q* ⇒ *'g inference set* **and**
Red-I-q :: *'q* ⇒ *'g set* ⇒ *'g inference set* **and**
Red-F-q :: *'q* ⇒ *'g set* ⇒ *'g set* **and**
$\mathcal{G}$*-F-q* :: *'q* ⇒ *'f* ⇒ *'g set* **and**
$\mathcal{G}$*-I-q* :: *'q* ⇒ *'f inference* ⇒ *'g inference set option*
\+ **fixes**
Inf-FL :: ‹(*'f* × *'l*) *inference set*›
assumes
Inf-F-to-Inf-FL:
‹ι_F ∈ *Inf-F* ⟹ *length* (*Ll* :: *'l list*) = *length* (*prems-of* ι_F) ⟹
∃ *L0*. *Infer* (*zip* (*prems-of* ι_F) *Ll*) (*concl-of* ι_F, *L0*) ∈ *Inf-FL*› **and**
Inf-FL-to-Inf-F: ‹ι_{FL} ∈ *Inf-FL* ⟹ *Infer* (*map fst* (*prems-of* ι_{FL})) (*fst* (*concl-of* ι_{FL})) ∈ *Inf-F*›
begin

definition *to-F* :: ‹(*'f* × *'l*) *inference* ⇒ *'f inference*› **where**
‹*to-F* ι_{FL} = *Infer* (*map fst* (*prems-of* ι_{FL})) (*fst* (*concl-of* ι_{FL}))›

abbreviation *Bot-FL* :: ‹(*'f* × *'l*) *set*› **where**
‹*Bot-FL* ≡ *Bot-F* × *UNIV*›

abbreviation $\mathcal{G}$*-F-L-q* :: ‹*'q* ⇒ (*'f* × *'l*) ⇒ *'g set*› **where**
‹$\mathcal{G}$*-F-L-q q CL* ≡ $\mathcal{G}$*-F-q q* (*fst CL*)›

abbreviation $\mathcal{G}$*-I-L-q* :: ‹*'q* ⇒ (*'f* × *'l*) *inference* ⇒ *'g inference set option*› **where**
‹$\mathcal{G}$*-I-L-q q* ι_{FL} ≡ $\mathcal{G}$*-I-q q* (*to-F* ι_{FL})›

abbreviation $\mathcal{G}$*-Fset-L-q* :: *'q* ⇒ (*'f* × *'l*) *set* ⇒ *'g set* **where**
$\mathcal{G}$*-Fset-L-q q N* ≡ ⋃ ($\mathcal{G}$*-F-L-q q* ‘ *N*)

definition *Red-I-*$\mathcal{G}$*-L-q* :: *'q* ⇒ (*'f* × *'l*) *set* ⇒ (*'f* × *'l*) *inference set* **where**
Red-I-$\mathcal{G}$*-L-q q N* =
{ι ∈ *Inf-FL*. ($\mathcal{G}$*-I-L-q q* ι ≠ *None* ∧ *the* ($\mathcal{G}$*-I-L-q q* ι) ⊆ *Red-I-q q* ($\mathcal{G}$*-Fset-L-q q N*))
∨ ($\mathcal{G}$*-I-L-q q* ι = *None* ∧ $\mathcal{G}$*-F-L-q q* (*concl-of* ι) ⊆ $\mathcal{G}$*-Fset-L-q q N* ∪ *Red-F-q q* ($\mathcal{G}$*-Fset-L-q q N*))}

abbreviation *Red-I-*$\mathcal{G}$*-L* :: (*'f* × *'l*) *set* ⇒ (*'f* × *'l*) *inference set* **where**
Red-I-$\mathcal{G}$*-L N* ≡ (⋂ *q* ∈ *Q*. *Red-I-*$\mathcal{G}$*-L-q q N*)

abbreviation *entails-*$\mathcal{G}$*-L-q* :: *'q* ⇒ (*'f* × *'l*) *set* ⇒ (*'f* × *'l*) *set* ⇒ *bool* **where**
entails-$\mathcal{G}$*-L-q q N1 N2* ≡ *entails-q q* ($\mathcal{G}$*-Fset-L-q q N1*) ($\mathcal{G}$*-Fset-L-q q N2*)

lemma *lifting-q*:
assumes *q* ∈ *Q*
shows *labeled-tiebreaker-lifting Bot-F Inf-F Bot-G* (*entails-q q*) (*Inf-G-q q*) (*Red-I-q q*)
(*Red-F-q q*) ($\mathcal{G}$*-F-q q*) ($\mathcal{G}$*-I-q q*) (λg *Cl Cl'*. *False*) *Inf-FL*
⟨*proof*⟩

lemma *lifted-q*:
assumes *q-in*: *q* ∈ *Q*

shows *standard-lifting Inf-FL Bot-G* (*Inf-G-q q*) (*entails-q q*) (*Red-I-q q*) (*Red-F-q q*)
Bot-FL ($\mathcal{G}$*-F-L-q q*) ($\mathcal{G}$*-I-L-q q*)
⟨*proof*⟩

lemma *ord-fam-lifted-q*:
assumes *q-in*: $q \in Q$
shows *tiebreaker-lifting Bot-FL Inf-FL Bot-G* (*entails-q q*) (*Inf-G-q q*) (*Red-I-q q*)
(*Red-F-q q*) ($\mathcal{G}$*-F-L-q q*) ($\mathcal{G}$*-I-L-q q*) (λg *Cl Cl'*. *False*)
⟨*proof*⟩

definition *Red-F-*$\mathcal{G}$*-empty-L-q* :: $'q \Rightarrow ('f \times 'l)$ *set* $\Rightarrow ('f \times 'l)$ *set* **where**
Red-F-$\mathcal{G}$*-empty-L-q q N* = {*C*. ∀ *D* ∈ $\mathcal{G}$*-F-L-q q C*. *D* ∈ *Red-F-q q* ($\mathcal{G}$*-Fset-L-q q N*) ∨
(∃ *E* ∈ *N*. *False* ∧ *D* ∈ $\mathcal{G}$*-F-L-q q E*)}

abbreviation *Red-F-*$\mathcal{G}$*-empty-L* :: $('f \times 'l)$ *set* $\Rightarrow ('f \times 'l)$ *set* **where**
Red-F-$\mathcal{G}$*-empty-L N* ≡ ($\bigcap$ *q* ∈ *Q*. *Red-F-*$\mathcal{G}$*-empty-L-q q N*)

lemma *all-lifted-red-crit*:
assumes *q-in*: $q \in Q$
shows *calculus Bot-FL Inf-FL* (*entails-*$\mathcal{G}$*-L-q q*) (*Red-I-*$\mathcal{G}$*-L-q q*) (*Red-F-*$\mathcal{G}$*-empty-L-q q*)
⟨*proof*⟩

lemma *all-lifted-cons-rel*:
assumes *q-in*: $q \in Q$
shows *consequence-relation Bot-FL* (*entails-*$\mathcal{G}$*-L-q q*)
⟨*proof*⟩

sublocale *consequence-relation-family Bot-FL Q entails-*$\mathcal{G}$*-L-q*
⟨*proof*⟩

sublocale *intersection-calculus Bot-FL Inf-FL Q entails-*$\mathcal{G}$*-L-q Red-I-*$\mathcal{G}$*-L-q Red-F-*$\mathcal{G}$*-empty-L-q*
⟨*proof*⟩

lemma *in-Inf-FL-imp-to-F-in-Inf-F*: $\iota \in$ *Inf-FL* $\Longrightarrow$ *to-F* $\iota \in$ *Inf-F*
⟨*proof*⟩

lemma *in-Inf-from-imp-to-F-in-Inf-from*: $\iota \in$ *Inf-from N* $\Longrightarrow$ *to-F* $\iota \in$ *no-labels.Inf-from* (*fst* ‘ *N*)
⟨*proof*⟩

notation *no-labels.entails-*$\mathcal{G}$ (**infix** ‹$\models\cap\mathcal{G}$› *50*)

abbreviation *entails-*$\mathcal{G}$*-L* :: $('f \times 'l)$ *set* $\Rightarrow ('f \times 'l)$ *set* $\Rightarrow$ *bool* (**infix** ‹$\models\cap\mathcal{G}L$› *50*) **where**
($\models\cap\mathcal{G}L$) ≡ *entails*

lemmas *entails-*$\mathcal{G}$*-L-def* = *entails-def*

lemma *labeled-entailment-lifting*: *NL1* $\models\cap\mathcal{G}L$ *NL2* $\longleftrightarrow$ *fst* ‘ *NL1* $\models\cap\mathcal{G}$ *fst* ‘ *NL2*
⟨*proof*⟩

lemma *red-inf-impl*: $\iota \in$ *Red-I NL* $\Longrightarrow$ *to-F* $\iota \in$ *no-labels.Red-I-*$\mathcal{G}$ (*fst* ‘ *NL*)
⟨*proof*⟩

lemma *labeled-family-saturation-lifting*: *saturated NL* $\Longrightarrow$ *no-labels.saturated* (*fst* ‘ *NL*)

⟨*proof*⟩

theorem *labeled-static-ref*:
assumes *calc*: *statically-complete-calculus Bot-F Inf-F* (⊨∩𝒢) *no-labels.Red-I-𝒢*
no-labels.Red-F-𝒢-empty
shows *statically-complete-calculus Bot-FL Inf-FL* (⊨∩𝒢*L*) *Red-I Red-F*
⟨*proof*⟩

end

end

6 Given Clause Prover Architectures

This section covers all the results presented in the section 4 of the report. This is where abstract architectures of provers are defined and proven dynamically refutationally complete.

theory *Given-Clause-Architectures*
imports
Lambda-Free-RPOs.Lambda-Free-Util
Labeled-Lifting-to-Non-Ground-Calculi
begin

6.1 Basis of the Given Clause Prover Architectures

locale *given-clause-basis* = *std?*: *labeled-lifting-intersection Bot-F Inf-F Bot-G Q*
entails-q Inf-G-q Red-I-q Red-F-q 𝒢-F-q 𝒢-I-q
$\{\iota_{FL}$ :: ('*f* × '*l*) *inference*. *Infer* (*map fst* (*prems-of* ι_{FL})) (*fst* (*concl-of* ι_{FL})) ∈ *Inf-F*$\}$
for
Bot-F :: '*f set*
and *Inf-F* :: '*f inference set*
and *Bot-G* :: '*g set*
and *Q* :: '*q set*
and *entails-q* :: '*q* ⇒ '*g set* ⇒ '*g set* ⇒ *bool*
and *Inf-G-q* :: ‹'*q* ⇒ '*g inference set*›
and *Red-I-q* :: '*q* ⇒ '*g set* ⇒ '*g inference set*
and *Red-F-q* :: '*q* ⇒ '*g set* ⇒ '*g set*
and *𝒢-F-q* :: '*q* ⇒ '*f* ⇒ '*g set*
and *𝒢-I-q* :: '*q* ⇒ '*f inference* ⇒ '*g inference set option*
\+ **fixes**
Equiv-F :: '*f* ⇒ '*f* ⇒ *bool* (**infix** ‹≐› *50*) **and**
Prec-F :: '*f* ⇒ '*f* ⇒ *bool* (**infix** ‹≺·› *50*) **and**
Prec-L :: '*l* ⇒ '*l* ⇒ *bool* (**infix** ‹⊏*L*› *50*) **and**
active :: '*l*
assumes
equiv-equiv-F: *equivp* (≐) **and**
wf-prec-F: *wfp* (≺·) *transp* (≺·) **and**
wf-prec-L: *wfp* (⊏*L*) *transp* (⊏*L*) **and**
compat-equiv-prec: *C1* ≐ *D1* ⟹ *C2* ≐ *D2* ⟹ *C1* ≺· *C2* ⟹ *D1* ≺· *D2* **and**
equiv-F-grounding: *q* ∈ *Q* ⟹ *C1* ≐ *C2* ⟹ *𝒢-F-q q C1* ⊆ *𝒢-F-q q C2* **and**
prec-F-grounding: *q* ∈ *Q* ⟹ *C2* ≺· *C1* ⟹ *𝒢-F-q q C1* ⊆ *𝒢-F-q q C2* **and**
active-minimal: *l2* ≠ *active* ⟹ *active* ⊏*L* *l2* **and**
at-least-two-labels: ∃ *l2*. *active* ⊏*L* *l2* **and**
static-ref-comp: *statically-complete-calculus Bot-F Inf-F* (⊨∩𝒢)

no-labels.Red-I-$\mathcal{G}$ no-labels.Red-F-$\mathcal{G}$-empty
begin

abbreviation *Inf-FL* :: $('f \times 'l)$ *inference set* **where**
Inf-FL $\equiv \{\iota_{FL}.$ *Infer* (*map fst* (*prems-of* ι_{FL})) (*fst* (*concl-of* ι_{FL})) $\in$ *Inf-F*$\}$

abbreviation *Prec-eq-F* :: $'f \Rightarrow 'f \Rightarrow$ *bool* (**infix** $\langle\preceq\cdot\rangle$ *50*) **where**
$C \preceq\cdot D \equiv C \doteq D \vee C \prec\cdot D$

definition *Prec-FL* :: $('f \times 'l) \Rightarrow ('f \times 'l) \Rightarrow$ *bool* (**infix** $\langle\sqsubset\rangle$ *50*) **where**
Cl1 $\sqsubset$ *Cl2* $\longleftrightarrow$ *fst Cl1* $\prec\cdot$ *fst Cl2* $\vee$ (*fst Cl1* $\doteq$ *fst Cl2* $\wedge$ *snd Cl1* $\sqsubset L$ *snd Cl2*)

lemma *irrefl-prec-F*: $\neg\ C \prec\cdot C$
$\langle$*proof*$\rangle$

lemma *trans-prec-F*: $C1 \prec\cdot C2 \Longrightarrow C2 \prec\cdot C3 \Longrightarrow C1 \prec\cdot C3$
$\langle$*proof*$\rangle$

lemma *wf-prec-FL*: *wfp* $(\sqsubset)$ *transp* $(\sqsubset)$
$\langle$*proof*$\rangle$

definition *active-subset* :: $('f \times 'l)$ *set* $\Rightarrow$ $('f \times 'l)$ *set* **where**
active-subset $M = \{CL \in M.$ *snd* $CL =$ *active*$\}$

definition *passive-subset* :: $('f \times 'l)$ *set* $\Rightarrow$ $('f \times 'l)$ *set* **where**
passive-subset $M = \{CL \in M.$ *snd* $CL \neq$ *active*$\}$

lemma *active-subset-insert*[*simp*]:
active-subset (*insert Cl N*) = (*if snd Cl = active then* $\{Cl\}$ *else* $\{\}$) $\cup$ *active-subset N*
$\langle$*proof*$\rangle$

lemma *active-subset-union*[*simp*]: *active-subset* $(M \cup N)$ = *active-subset M* $\cup$ *active-subset N*
$\langle$*proof*$\rangle$

lemma *passive-subset-insert*[*simp*]:
passive-subset (*insert Cl N*) = (*if snd Cl* $\neq$ *active then* $\{Cl\}$ *else* $\{\}$) $\cup$ *passive-subset N*
$\langle$*proof*$\rangle$

lemma *passive-subset-union*[*simp*]: *passive-subset* $(M \cup N)$ = *passive-subset M* $\cup$ *passive-subset N*
$\langle$*proof*$\rangle$

sublocale *std?*: *statically-complete-calculus Bot-FL Inf-FL* $(\models\cap\mathcal{G}L)$ *Red-I Red-F*
$\langle$*proof*$\rangle$

lemma *labeled-tiebreaker-lifting*:
assumes *q-in*: $q \in Q$
shows *tiebreaker-lifting Bot-FL Inf-FL Bot-G* (*entails-q q*) (*Inf-G-q q*)
(*Red-I-q q*) (*Red-F-q q*) ($\mathcal{G}$-*F-L-q q*) ($\mathcal{G}$-*I-L-q q*) ($\lambda g.$ *Prec-FL*)
$\langle$*proof*$\rangle$

sublocale *lifting-intersection Inf-FL Bot-G Q Inf-G-q entails-q Red-I-q Red-F-q*
Bot-FL $\mathcal{G}$-*F-L-q* $\mathcal{G}$-*I-L-q* $\lambda g.$ *Prec-FL*
$\langle$*proof*$\rangle$

notation *derive* (**infix** $\langle\rhd L\rangle$ *50*)

lemma *std-Red-I-eq*: $std.Red\text{-}I = Red\text{-}I\text{-}\mathcal{G}$
⟨*proof*⟩

lemma *std-Red-F-eq*: $std.Red\text{-}F = Red\text{-}F\text{-}\mathcal{G}\text{-}empty$
⟨*proof*⟩

sublocale *statically-complete-calculus Bot-FL Inf-FL* $(\models\cap\mathcal{G}L)$ *Red-I Red-F*
⟨*proof*⟩

lemma *labeled-red-inf-eq-red-inf*:
assumes *i-in*: $\iota \in Inf\text{-}FL$
shows $\iota \in Red\text{-}I\ N \longleftrightarrow to\text{-}F\ \iota \in no\text{-}labels.Red\text{-}I\text{-}\mathcal{G}\ (fst\ `\ N)$
⟨*proof*⟩

lemma *red-labeled-clauses*:
assumes ‹$C \in no\text{-}labels.Red\text{-}F\text{-}\mathcal{G}\text{-}empty\ (fst\ `\ N) \lor$
$(\exists\, C' \in fst\ `\ N.\ C' \prec\cdot\ C) \lor (\exists\, (C', L') \in N.\ L' \sqsubset L\ L \land C' \preceq\cdot\ C)$›
shows ‹$(C, L) \in Red\text{-}F\ N$›
⟨*proof*⟩

end

6.2 Given Clause Procedure

locale *given-clause = given-clause-basis Bot-F Inf-F Bot-G Q entails-q Inf-G-q Red-I-q Red-F-q* $\mathcal{G}$*-F-q* $\mathcal{G}$*-I-q Equiv-F Prec-F Prec-L active*
for
Bot-F :: *'f set* **and**
Inf-F :: *'f inference set* **and**
Bot-G :: *'g set* **and**
Q :: *'q set* **and**
entails-q :: $'q \Rightarrow 'g\ set \Rightarrow 'g\ set \Rightarrow bool$ **and**
Inf-G-q :: ‹$'q \Rightarrow 'g\ inference\ set$› **and**
Red-I-q :: $'q \Rightarrow 'g\ set \Rightarrow 'g\ inference\ set$ **and**
Red-F-q :: $'q \Rightarrow 'g\ set \Rightarrow 'g\ set$ **and**
$\mathcal{G}$*-F-q* :: $'q \Rightarrow 'f \Rightarrow 'g\ set$ **and**
$\mathcal{G}$*-I-q* :: $'q \Rightarrow 'f\ inference \Rightarrow 'g\ inference\ set\ option$ **and**
Equiv-F :: $'f \Rightarrow 'f \Rightarrow bool$ (**infix** ‹$\doteq$› *50*) **and**
Prec-F :: $'f \Rightarrow 'f \Rightarrow bool$ (**infix** ‹$\prec\cdot$› *50*) **and**
Prec-L :: $'l \Rightarrow 'l \Rightarrow bool$ (**infix** ‹$\sqsubset L$› *50*) **and**
active :: *'l* +
assumes
inf-have-prems: $\iota F \in Inf\text{-}F \Longrightarrow prems\text{-}of\ \iota F \neq []$
begin

lemma *labeled-inf-have-prems*: $\iota \in Inf\text{-}FL \Longrightarrow prems\text{-}of\ \iota \neq []$
⟨*proof*⟩

inductive *step* :: $('f \times 'l)\ set \Rightarrow ('f \times 'l)\ set \Rightarrow bool$ (**infix** ‹$\leadsto GC$› *50*) **where**
process: $N1 = N \cup M \Longrightarrow N2 = N \cup M' \Longrightarrow M \subseteq Red\text{-}F\ (N \cup M') \Longrightarrow$
$active\text{-}subset\ M' = \{\} \Longrightarrow N1 \leadsto GC\ N2$
| *infer*: $N1 = N \cup \{(C, L)\} \Longrightarrow N2 = N \cup \{(C, active)\} \cup M \Longrightarrow L \neq active \Longrightarrow$
$active\text{-}subset\ M = \{\} \Longrightarrow$

no-labels.Inf-between (*fst* ' *active-subset* *N*) {*C*}
$\subseteq$ *no-labels.Red-I* (*fst* ' ($N \cup \{(C,\ active)\} \cup M$)) $\Longrightarrow$
N1 $\rightsquigarrow$*GC N2*

lemma *one-step-equiv*: *N1* $\rightsquigarrow$*GC N2* $\Longrightarrow$ *N1* $\rhd$*L N2*
⟨*proof*⟩

lemma *gc-to-red*: *chain* ($\rightsquigarrow$*GC*) *Ns* $\Longrightarrow$ *chain* ($\rhd$*L*) *Ns*
⟨*proof*⟩

lemma (**in**−) *all-ex-finite-set*: ($\forall$ (*j*::*nat*)$\in$\{*0*..<*m*\}. $\exists$ (*n*::*nat*). *P j n*) $\Longrightarrow$
($\forall$ *n1 n2*. $\forall$ *j*$\in$\{*0*..<*m*\}. *P j n1* $\longrightarrow$ *P j n2* $\longrightarrow$ *n1* = *n2*) $\Longrightarrow$ *finite* \{*n*. $\exists$ *j* $\in$ \{*0*..<*m*\}. *P j n*\} **for** *m*
P
⟨*proof*⟩

lemma *gc-fair*:
assumes
deriv: *chain* ($\rightsquigarrow$*GC*) *Ns* **and**
init-state: *active-subset* (*lhd Ns*) = {} **and**
final-state: *passive-subset* (*Liminf-llist Ns*) = {}
shows *fair Ns*
⟨*proof*⟩

theorem *gc-complete-Liminf*:
assumes
deriv: *chain* ($\rightsquigarrow$*GC*) *Ns* **and**
init-state: *active-subset* (*lhd Ns*) = {} **and**
final-state: *passive-subset* (*Liminf-llist Ns*) = {} **and**
b-in: *B* $\in$ *Bot-F* **and**
bot-entailed: *no-labels.entails-*$\mathcal{G}$ (*fst* ' *lhd Ns*) {*B*}
shows $\exists$ *BL* $\in$ *Bot-FL*. *BL* $\in$ *Liminf-llist Ns*
⟨*proof*⟩

theorem *gc-complete*:
assumes
deriv: *chain* ($\rightsquigarrow$*GC*) *Ns* **and**
init-state: *active-subset* (*lhd Ns*) = {} **and**
final-state: *passive-subset* (*Liminf-llist Ns*) = {} **and**
b-in: *B* $\in$ *Bot-F* **and**
bot-entailed: *no-labels.entails-*$\mathcal{G}$ (*fst* ' *lhd Ns*) {*B*}
shows $\exists$ *i*. *enat i* < *llength Ns* $\wedge$ ($\exists$ *BL* $\in$ *Bot-FL*. *BL* $\in$ *lnth Ns i*)
⟨*proof*⟩

end

6.3 Lazy Given Clause Procedure

locale *lazy-given-clause* = *given-clause-basis Bot-F Inf-F Bot-G Q entails-q Inf-G-q Red-I-q*
Red-F-q $\mathcal{G}$*-F-q* $\mathcal{G}$*-I-q Equiv-F Prec-F Prec-L active*
for
Bot-F :: *'f set* **and**
Inf-F :: *'f inference set* **and**
Bot-G :: *'g set* **and**

Q :: $'q$ *set* **and**
entails-q :: $'q \Rightarrow 'g$ *set* $\Rightarrow 'g$ *set* $\Rightarrow$ *bool* **and**
Inf-G-q :: ‹$'q \Rightarrow 'g$ *inference set*› **and**
Red-I-q :: $'q \Rightarrow 'g$ *set* $\Rightarrow 'g$ *inference set* **and**
Red-F-q :: $'q \Rightarrow 'g$ *set* $\Rightarrow 'g$ *set* **and**
$\mathcal{G}$*-F-q* :: $'q \Rightarrow 'f \Rightarrow 'g$ *set* **and**
$\mathcal{G}$*-I-q* :: $'q \Rightarrow 'f$ *inference* $\Rightarrow 'g$ *inference set option* **and**
Equiv-F :: $'f \Rightarrow 'f \Rightarrow$ *bool* (**infix** ‹$\doteq$› *50*) **and**
Prec-F :: $'f \Rightarrow 'f \Rightarrow$ *bool* (**infix** ‹$\prec\cdot$› *50*) **and**
Prec-L :: $'l \Rightarrow 'l \Rightarrow$ *bool* (**infix** ‹$\sqsubset L$› *50*) **and**
active :: $'l$
begin

inductive *step* :: $'f$ *inference set* $\times$ $('f \times 'l)$ *set* $\Rightarrow$
$'f$ *inference set* $\times$ $('f \times 'l)$ *set* $\Rightarrow$ *bool* (**infix** ‹$\rightsquigarrow LGC$› *50*) **where**
process: $N1 = N \cup M \Longrightarrow N2 = N \cup M' \Longrightarrow M \subseteq \mathit{Red\text{-}F}\ (N \cup M') \Longrightarrow$
active-subset $M' = \{\} \Longrightarrow (T,\ N1) \rightsquigarrow LGC\ (T,\ N2)$ |
schedule-infer: $T2 = T1 \cup T' \Longrightarrow N1 = N \cup \{(C,\ L)\} \Longrightarrow N2 = N \cup \{(C,\ \mathit{active})\} \Longrightarrow$
$L \neq \mathit{active} \Longrightarrow T' = \mathit{no\text{-}labels.Inf\text{-}between}\ (\mathit{fst}\ \text{‘}\ \mathit{active\text{-}subset}\ N)\ \{C\} \Longrightarrow$
$(T1,\ N1) \rightsquigarrow LGC\ (T2,\ N2)$ |
compute-infer: $T1 = T2 \cup \{\iota\} \Longrightarrow N2 = N1 \cup M \Longrightarrow \mathit{active\text{-}subset}\ M = \{\} \Longrightarrow$
$\iota \in \mathit{no\text{-}labels.Red\text{-}I}\ (\mathit{fst}\ \text{‘}\ (N1 \cup M)) \Longrightarrow (T1,\ N1) \rightsquigarrow LGC\ (T2,\ N2)$ |
delete-orphan-infers: $T1 = T2 \cup T' \Longrightarrow$
$T' \cap \mathit{no\text{-}labels.Inf\text{-}from}\ (\mathit{fst}\ \text{‘}\ \mathit{active\text{-}subset}\ N) = \{\} \Longrightarrow (T1,\ N) \rightsquigarrow LGC\ (T2,\ N)$

lemma *premise-free-inf-always-from*: $\iota \in \mathit{Inf\text{-}F} \Longrightarrow \mathit{prems\text{-}of}\ \iota = [] \Longrightarrow \iota \in \mathit{no\text{-}labels.Inf\text{-}from}\ N$
⟨*proof*⟩

lemma *one-step-equiv*: $(T1,\ N1) \rightsquigarrow LGC\ (T2,\ N2) \Longrightarrow N1 \rhd L\ N2$
⟨*proof*⟩

lemma *lgc-to-red*: *chain* $(\rightsquigarrow LGC)$ *Ns* $\Longrightarrow$ *chain* $(\rhd L)$ (*lmap snd Ns*)
⟨*proof*⟩

lemma *lgc-fair*:
assumes
deriv: *chain* $(\rightsquigarrow LGC)$ *Ns* **and**
init-state: *active-subset* (*snd* (*lhd Ns*)) $= \{\}$ **and**
final-state: *passive-subset* (*Liminf-llist* (*lmap snd Ns*)) $= \{\}$ **and**
no-prems-init: $\forall \iota \in \mathit{Inf\text{-}F}.\ \mathit{prems\text{-}of}\ \iota = [] \longrightarrow \iota \in \mathit{fst}\ (\mathit{lhd}\ \mathit{Ns})$ **and**
final-schedule: *Liminf-llist* (*lmap fst Ns*) $= \{\}$
shows *fair* (*lmap snd Ns*)
⟨*proof*⟩

theorem *lgc-complete-Liminf*:
assumes
deriv: *chain* $(\rightsquigarrow LGC)$ *Ns* **and**
init-state: *active-subset* (*snd* (*lhd Ns*)) $= \{\}$ **and**
final-state: *passive-subset* (*Liminf-llist* (*lmap snd Ns*)) $= \{\}$ **and**
no-prems-init: $\forall \iota \in \mathit{Inf\text{-}F}.\ \mathit{prems\text{-}of}\ \iota = [] \longrightarrow \iota \in \mathit{fst}\ (\mathit{lhd}\ \mathit{Ns})$ **and**
final-schedule: *Liminf-llist* (*lmap fst Ns*) $= \{\}$ **and**
b-in: $B \in \mathit{Bot\text{-}F}$ **and**
bot-entailed: *no-labels.entails-*$\mathcal{G}$ (*fst* ‘ *snd* (*lhd Ns*)) $\{B\}$

shows $\exists BL \in \mathit{Bot\text{-}FL}.\ BL \in \mathit{Liminf\text{-}llist}\ (\mathit{lmap\ snd\ Ns})$
$\langle proof \rangle$

theorem *lgc-complete*:
assumes
deriv: $\mathit{chain}\ (\leadsto LGC)\ \mathit{Ns}$ **and**
init-state: $\mathit{active\text{-}subset}\ (\mathit{snd}\ (\mathit{lhd\ Ns})) = \{\}$ **and**
final-state: $\mathit{passive\text{-}subset}\ (\mathit{Liminf\text{-}llist}\ (\mathit{lmap\ snd\ Ns})) = \{\}$ **and**
no-prems-init: $\forall \iota \in \mathit{Inf\text{-}F}.\ \mathit{prems\text{-}of}\ \iota = [] \longrightarrow \iota \in \mathit{fst}\ (\mathit{lhd\ Ns})$ **and**
final-schedule: $\mathit{Liminf\text{-}llist}\ (\mathit{lmap\ fst\ Ns}) = \{\}$ **and**
b-in: $B \in \mathit{Bot\text{-}F}$ **and**
bot-entailed: $\mathit{no\text{-}labels.entails\text{-}}\mathcal{G}\ (\mathit{fst}\ `\ \mathit{snd}\ (\mathit{lhd\ Ns}))\ \{B\}$
shows $\exists i.\ \mathit{enat}\ i < \mathit{llength\ Ns} \wedge (\exists BL \in \mathit{Bot\text{-}FL}.\ BL \in \mathit{snd}\ (\mathit{lnth\ Ns}\ i))$
$\langle proof \rangle$

end

end