Relational Characterisations of Paths

Walter Guttmann and Peter Höfner

March 17, 2025

Abstract

Binary relations are one of the standard ways to encode, characterise and reason about graphs. Relation algebras provide equational axioms for a large fragment of the calculus of binary relations. Although relations are standard tools in many areas of mathematics and computing, researchers usually fall back to point-wise reasoning when it comes to arguments about paths in a graph. We present a purely algebraic way to specify different kinds of paths in Kleene relation algebras, which are relation algebras equipped with an operation for reflexive transitive closure. We study the relationship between paths with a designated root vertex and paths without such a vertex. Since we stay in first-order logic this development helps with mechanising proofs. To demonstrate the applicability of the algebraic framework we verify the correctness of three basic graph algorithms.

Contents

Overview

A path in a graph can be defined as a connected subgraph of edges where each vertex has at most one incoming edge and at most one outgoing edge [3, 12]. We develop a theory of paths based on this representation and use it for algorithm verification. All reasoning is done in variants of relation algebras and Kleene algebras [8, 9, 11].

Section 1 presents fundamental results that hold in relation algebras. Relation-algebraic characterisations of various kinds of paths are introduced and compared in Section 2. We extend this to paths with a designated root in Section 3. Section 4 verifies the correctness of a few basic graph algorithms.

These Isabelle/HOL theories formally verify results in [2]. See this paper for further details and related work.

1 (More) Relation Algebra

This theory presents fundamental properties of relation algebras, which are not present in the AFP entry on relation algebras but could be integrated there [1]. Many theorems concern vectors and points.

theory *More-Relation-Algebra*

imports *Relation-Algebra.Relation-Algebra-RTC*
Relation-Algebra.Relation-Algebra-Functions

begin

unbundle *no trancl-syntax*

context *relation-algebra*
begin

notation
  converse (‹(-^{T})› [*102*] *101*)

abbreviation *bijective*
  where *bijective* $x \equiv$ *is-inj* $x \wedge$ *is-sur* x

abbreviation *reflexive*
  where *reflexive* $R \equiv 1' \leq R$

abbreviation *symmetric*
  where *symmetric* $R \equiv R = R^T$

abbreviation *transitive*
  where *transitive* $R \equiv R;R \leq R$

General theorems

lemma *x-leq-triple-x*:
$x \leq x;x^T;x$
⟨*proof*⟩

lemma *inj-triple*:
assumes *is-inj* x
shows $x = x;x^T;x$
⟨*proof*⟩

lemma *p-fun-triple*:
assumes *is-p-fun* x
shows $x = x;x^T;x$
⟨*proof*⟩

lemma *loop-backward-forward*:
$x^T \leq -(1') + x$
⟨*proof*⟩

lemma *inj-sur-semi-swap*:
assumes *is-sur* z
and *is-inj* x
shows $z \leq y;x \Longrightarrow x \leq y^T;z$
⟨*proof*⟩

lemma *inj-sur-semi-swap-short*:
assumes *is-sur* z
and *is-inj* x
shows $z \leq y^T;x \Longrightarrow x \leq y;z$
⟨*proof*⟩

lemma *bij-swap*:
assumes *bijective* z
and *bijective* x
shows $z \leq y^T;x \longleftrightarrow x \leq y;z$
⟨*proof*⟩

The following result is [10, Proposition 4.2.2(iv)].

lemma *ss422iv*:
assumes *is-p-fun* y
and $x \leq y$
and $y;1 \leq x;1$
shows $x = y$
⟨*proof*⟩

The following results are variants of [10, Proposition 4.2.3].

lemma *ss423conv*:
assumes *bijective* x
shows $x ; y \leq z \longleftrightarrow y \leq x^T ; z$

⟨*proof*⟩

lemma *ss423bij*:
assumes *bijective x*
shows $y ; x^T \leq z \longleftrightarrow y \leq z ; x$
⟨*proof*⟩

lemma *inj-distr*:
assumes *is-inj z*
shows $(x \cdot y);z = (x;z) \cdot (y;z)$
⟨*proof*⟩

lemma *test-converse*:
$x \cdot 1' = x^T \cdot 1'$
⟨*proof*⟩

lemma *injective-down-closed*:
assumes *is-inj x*
and $y \leq x$
shows *is-inj y*
⟨*proof*⟩

lemma *injective-sup*:
assumes *is-inj t*
and $e;t^T \leq 1'$
and *is-inj e*
shows *is-inj* $(t + e)$
⟨*proof*⟩

Some (more) results about vectors

lemma *vector-meet-comp*:
assumes *is-vector v*
and *is-vector w*
shows $v;w^T = v \cdot w^T$
⟨*proof*⟩

lemma *vector-meet-comp'*:
assumes *is-vector v*
shows $v;v^T = v \cdot v^T$
⟨*proof*⟩

lemma *vector-meet-comp-x*:
$x;1;x^T = x;1 \cdot 1;x^T$
⟨*proof*⟩

lemma *vector-meet-comp-x'*:
$x;1;x = x;1 \cdot 1;x$
⟨*proof*⟩

lemma *vector-prop1*:
assumes *is-vector* v
shows $-v^T;v = 0$
⟨*proof*⟩

The following results and a number of others in this theory are from [5].

lemma *ee*:
assumes *is-vector* v
and $e \leq v;-v^T$
shows $e;e = 0$
⟨*proof*⟩

lemma *et*:
assumes *is-vector* v
and $e \leq v;-v^T$
and $t \leq v;v^T$
shows $e;t = 0$
and $e;t^T = 0$
⟨*proof*⟩

Some (more) results about points

definition *point*
where *point* $x \equiv$ *is-vector* $x \wedge$ *bijective* x

lemma *point-swap*:
assumes *point* p
and *point* q
shows $p \leq x;q \longleftrightarrow q \leq x^T;p$
⟨*proof*⟩

Some (more) results about singletons

abbreviation *singleton*
where *singleton* $x \equiv$ *bijective* $(x;1) \wedge$ *bijective* $(x^T;1)$

lemma *singleton-injective*:
assumes *singleton* x
shows *is-inj* x
⟨*proof*⟩

lemma *injective-inv*:
assumes *is-vector* v
and *singleton* e
and $e \leq v;-v^T$
and $t \leq v;v^T$
and *is-inj* t
shows *is-inj* $(t + e)$
⟨*proof*⟩

lemma *singleton-is-point*:

assumes *singleton p*
shows *point* $(p;1)$
⟨*proof*⟩

lemma *singleton-transp*:
assumes *singleton p*
shows *singleton* (p^T)
⟨*proof*⟩

lemma *point-to-singleton*:
assumes *singleton p*
shows *singleton* $(1'\cdot p;p^T)$
⟨*proof*⟩

lemma *singleton-singletonT*:
assumes *singleton p*
shows $p;p^T \leq 1'$
⟨*proof*⟩

Minimality

abbreviation *minimum*
where *minimum* $x\ v \equiv v \cdot -(x^T;v)$

Regressively finite

abbreviation *regressively-finite*
where *regressively-finite* $x \equiv \forall\, v\ .\ \textit{is-vector}\ v \wedge v \leq x^T;v \longrightarrow v = 0$

lemma *regressively-finite-minimum*:
regressively-finite $R \Longrightarrow$ *is-vector* $v \Longrightarrow v \neq 0 \Longrightarrow$ *minimum* $R\ v \neq 0$
⟨*proof*⟩

lemma *regressively-finite-irreflexive*:
assumes *regressively-finite x*
shows $x \leq -1'$
⟨*proof*⟩

end

1.1 Relation algebras satisfying the Tarski rule

class *relation-algebra-tarski* = *relation-algebra* +
assumes *tarski*: $x \neq 0 \longleftrightarrow 1;x;1 = 1$
begin

Some (more) results about points

lemma *point-equations*:
assumes *is-point p*
shows $p;1=p$
and $1;p=1$

and $p^T;1=1$
and $1;p^T=p^T$
⟨*proof*⟩

The following result is [10, Proposition 2.4.5(i)].

lemma *point-singleton*:
assumes *is-point p*
and *is-vector v*
and $v \neq 0$
and $v \leq p$
shows $v = p$
⟨*proof*⟩

lemma *point-not-equal-aux*:
assumes *is-point p*
and *is-point q*
shows $p{\neq}q \longleftrightarrow p \cdot -q \neq 0$
⟨*proof*⟩

The following result is part of [10, Proposition 2.4.5(ii)].

lemma *point-not-equal*:
assumes *is-point p*
and *is-point q*
shows $p{\neq}q \longleftrightarrow p{\leq}-q$
and $p{\leq}-q \longleftrightarrow p;q^T \leq -1'$
and $p;q^T \leq -1' \longleftrightarrow p^T;q \leq 0$
⟨*proof*⟩

lemma *point-is-point*:
point x $\longleftrightarrow$ *is-point x*
⟨*proof*⟩

lemma *point-in-vector-or-complement*:
assumes *point p*
and *is-vector v*
shows $p \leq v \vee p \leq -v$
⟨*proof*⟩

lemma *point-in-vector-or-complement-iff*:
assumes *point p*
and *is-vector v*
shows $p \leq v \longleftrightarrow \neg(p \leq -v)$
⟨*proof*⟩

lemma *different-points-consequences*:
assumes *point p*
and *point q*
and $p{\neq}q$
shows $p^T;-q=1$

and $-q^T;p=1$
and $-(p^T;-q)=0$
and $-(-q^T;p)=0$
⟨*proof*⟩

Some (more) results about singletons

lemma *singleton-pq*:
assumes *point p*
and *point q*
shows *singleton* $(p;q^T)$
⟨*proof*⟩

lemma *singleton-equal-aux*:
assumes *singleton p*
and *singleton q*
and $q \leq p$
shows $p \leq q;1$
⟨*proof*⟩

lemma *singleton-equal*:
assumes *singleton p*
and *singleton q*
and $q \leq p$
shows $q=p$
⟨*proof*⟩

lemma *singleton-nonsplit*:
assumes *singleton p*
and $x \leq p$
shows $x=0 \vee x=p$
⟨*proof*⟩

lemma *singleton-nonzero*:
assumes *singleton p*
shows $p \neq 0$
⟨*proof*⟩

lemma *singleton-sum*:
assumes *singleton p*
shows $p \leq x+y \longleftrightarrow (p \leq x \vee p \leq y)$
⟨*proof*⟩

lemma *singleton-iff*:
singleton x $\longleftrightarrow x \neq 0 \wedge x^T;1;x + x;1;x^T \leq 1'$
⟨*proof*⟩

lemma *singleton-not-atom-in-relation-algebra-tarski*:
assumes $p \neq 0$
and $\forall x . x \leq p \longrightarrow x=0 \vee x=p$

shows *singleton p*
nitpick [*expect=genuine*] ⟨*proof*⟩

end

1.2 Relation algebras satisfying the point axiom

class *relation-algebra-point* = *relation-algebra* +
assumes *point-axiom*: $x \neq 0 \longrightarrow (\exists\, y\ z\ .\ point\ y \wedge point\ z \wedge y;z^T \leq x)$
begin

Some (more) results about points

lemma *point-exists*:
$\exists\, x\ .\ point\ x$
⟨*proof*⟩

lemma *point-below-vector*:
assumes *is-vector v*
and $v \neq 0$
shows $\exists\, x\ .\ point\ x \wedge x \leq v$
⟨*proof*⟩

end

class *relation-algebra-tarski-point* = *relation-algebra-tarski* +
relation-algebra-point
begin

lemma *atom-is-singleton*:
assumes $p \neq 0$
and $\forall\, x\ .\ x \leq p \longrightarrow x=0 \vee x=p$
shows *singleton p*
⟨*proof*⟩

lemma *singleton-iff-atom*:
$singleton\ p \longleftrightarrow p \neq 0 \wedge (\forall\, x\ .\ x \leq p \longrightarrow x=0 \vee x=p)$
⟨*proof*⟩

lemma *maddux-tarski*:
assumes $x \neq 0$
shows $\exists\, y\ .\ y \neq 0 \wedge y \leq x \wedge is\text{-}p\text{-}fun\ y$
⟨*proof*⟩

Intermediate Point Theorem [10, Proposition 2.4.8]

lemma *intermediate-point-theorem*:
assumes *point p*
and *point r*
shows $p \leq x;y;r \longleftrightarrow (\exists\, q\ .\ point\ q \wedge p \leq x;q \wedge q \leq y;r)$
⟨*proof*⟩

end

context *relation-algebra*
begin

lemma *unfoldl-inductl-implies-unfoldr*:
  assumes $\bigwedge x.\ 1' + x;(rtc\ x) \leq rtc\ x$
    and $\bigwedge x\ y\ z.\ x{+}y;z \leq z \Longrightarrow rtc(y);x \leq z$
   shows $1' + rtc(x);x \leq rtc\ x$
⟨*proof*⟩

lemma *star-transpose-swap*:
  assumes $\bigwedge x.\ 1' + x;(rtc\ x) \leq rtc\ x$
    and $\bigwedge x\ y\ z.\ x{+}y;z \leq z \Longrightarrow rtc(y);x \leq z$
   shows $rtc(x^T) = (rtc\ x)^T$
⟨*proof*⟩

lemma *unfoldl-inductl-implies-inductr*:
  assumes $\bigwedge x.\ 1' + x;(rtc\ x) \leq rtc\ x$
    and $\bigwedge x\ y\ z.\ x{+}y;z \leq z \Longrightarrow rtc(y);x \leq z$
   shows $x{+}z;y \leq z \Longrightarrow x;rtc(y) \leq z$
⟨*proof*⟩

end

context *relation-algebra-rtc*
begin

abbreviation *tc* (‹(-$^+$)› [*101*] *100*) **where** $tc\ x \equiv x;x^\star$

abbreviation *is-acyclic*
  where $is\text{-}acyclic\ x \equiv x^+ \leq -1'$

General theorems

lemma *star-denest-10*:
  assumes $x;y{=}0$
   shows $(x{+}y)^\star = y;y^\star;x^\star{+}x^\star$
⟨*proof*⟩

lemma *star-star-plus*:
  $x^\star + y^\star = x^+ + y^\star$
⟨*proof*⟩

The following two lemmas are from [6].

lemma *cancel-separate*:
  assumes $x\ ;\ y \leq 1'$
  shows $x^\star\ ;\ y^\star \leq x^\star + y^\star$

⟨*proof*⟩

lemma *cancel-separate-inj-converse*:
assumes *is-inj* x
shows $x^\star ; x^{T\star} = x^\star + x^{T\star}$
⟨*proof*⟩

lemma *cancel-separate-p-fun-converse*:
assumes *is-p-fun* x
shows $x^{T\star} ; x^\star = x^\star + x^{T\star}$
⟨*proof*⟩

lemma *cancel-separate-converse-idempotent*:
assumes *is-inj* x
and *is-p-fun* x
shows $(x^\star + x^{T\star});(x^\star + x^{T\star}) = x^\star + x^{T\star}$
⟨*proof*⟩

lemma *triple-star*:
assumes *is-inj* x
and *is-p-fun* x
shows $x^\star;x^{T\star};x^\star = x^\star + x^{T\star}$
⟨*proof*⟩

lemma *inj-xxts*:
assumes *is-inj* x
shows $x;x^{T\star} \leq x^\star + x^{T\star}$
⟨*proof*⟩

lemma *plus-top*:
$x^+;1 = x;1$
⟨*proof*⟩

lemma *top-plus*:
$1;x^+ = 1;x$
⟨*proof*⟩

lemma *plus-conv*:
$(x^+)^T = x^{T+}$
⟨*proof*⟩

lemma *inj-implies-step-forwards-backwards*:
assumes *is-inj* x
shows $x^\star;(x^+ \cdot 1');1 \leq x^T;1$
⟨*proof*⟩

Acyclic relations

The following result is from [4].

lemma *acyclic-inv*:

assumes *is-acyclic t*
and *is-vector v*
and $e \leq v;-v^T$
and $t \leq v;v^T$
shows *is-acyclic* $(t + e)$
⟨*proof*⟩

lemma *acyclic-single-step*:
assumes *is-acyclic x*
shows $x \leq -1'$
⟨*proof*⟩

lemma *acyclic-reachable-points*:
assumes *is-point p*
and *is-point q*
and $p \leq x;q$
and *is-acyclic x*
shows $p \neq q$
⟨*proof*⟩

lemma *acyclic-trans*:
assumes *is-acyclic x*
shows $x \leq -(x^{T+})$
⟨*proof*⟩

lemma *acyclic-trans′*:
assumes *is-acyclic x*
shows $x^\star \leq -(x^{T+})$
⟨*proof*⟩

Regressively finite

lemma *regressively-finite-acyclic*:
assumes *regressively-finite x*
shows *is-acyclic x*
⟨*proof*⟩

notation *power* (**infixr** ‹↑› *80*)

lemma *power-suc-below-plus*:
$x \uparrow Suc\ n \leq x^+$
⟨*proof*⟩

end

class *relation-algebra-rtc-tarski* = *relation-algebra-rtc* + *relation-algebra-tarski*
begin

lemma *point-loop-not-acyclic*:
assumes *is-point p*

and $p \leq x \uparrow Suc\ n\ ;\ p$
shows $\neg$ *is-acyclic x*
⟨*proof*⟩

end

class *relation-algebra-rtc-point* = *relation-algebra-rtc* + *relation-algebra-point*

class *relation-algebra-rtc-tarski-point* = *relation-algebra-rtc-tarski* + *relation-algebra-rtc-point* +
relation-algebra-tarski-point

Finite graphs: the axiom says the algebra has finitely many elements. This means the relations have a finite base set.

class *relation-algebra-rtc-tarski-point-finite* = *relation-algebra-rtc-tarski-point* + *finite*
begin

For a finite acyclic relation, the powers eventually vanish.

lemma *acyclic-power-vanishes*:
assumes *is-acyclic x*
shows $\exists n\ .\ x \uparrow Suc\ n = 0$
⟨*proof*⟩

Hence finite acyclic relations are regressively finite.

lemma *acyclic-regressively-finite*:
assumes *is-acyclic x*
shows *regressively-finite x*
⟨*proof*⟩

lemma *acyclic-is-regressively-finite*:
is-acyclic x $\longleftrightarrow$ *regressively-finite x*
⟨*proof*⟩

end

end

2 Relational Characterisation of Paths

This theory provides the relation-algebraic characterisations of paths, as defined in Sections 3–5 of [2].

theory *Paths*

imports *More-Relation-Algebra*

begin

context *relation-algebra-tarski*
begin

lemma *path-concat-aux-0*:
  assumes *is-vector* v
    and $v \neq 0$
    and $w;v^T \leq x$
    and $v;z \leq y$
  shows $w;1;z \leq x;y$
⟨*proof*⟩

end

2.1 Consequences without the Tarski rule

context *relation-algebra-rtc*
begin

Definitions for path classifications

abbreviation *connected*
  where *connected* $x \equiv x;1;x \leq x^\star + x^{T\star}$

abbreviation *many-strongly-connected*
  where *many-strongly-connected* $x \equiv x^\star = x^{T\star}$

abbreviation *one-strongly-connected*
  where *one-strongly-connected* $x \equiv x^T;1;x^T \leq x^\star$

definition *path*
  where *path* $x \equiv$ *connected* $x \wedge$ *is-p-fun* $x \wedge$ *is-inj* x

abbreviation *cycle*
  where *cycle* $x \equiv$ *path* $x \wedge$ *many-strongly-connected* x

abbreviation *start-points*
  where *start-points* $x \equiv x;1 \cdot -(x^T;1)$

abbreviation *end-points*
  where *end-points* $x \equiv x^T;1 \cdot -(x;1)$

abbreviation *no-start-points*
  where *no-start-points* $x \equiv x;1 \leq x^T;1$

abbreviation *no-end-points*
  where *no-end-points* $x \equiv x^T;1 \leq x;1$

abbreviation *no-start-end-points*
  where *no-start-end-points* $x \equiv x;1 = x^T;1$

abbreviation *has-start-points*

where *has-start-points* $x \equiv 1 = -(1;x);x;1$

abbreviation *has-end-points*
where *has-end-points* $x \equiv 1 = 1;x;-(x;1)$

abbreviation *has-start-end-points*
where *has-start-end-points* $x \equiv 1 = -(1;x);x;1 \cdot 1;x;-(x;1)$

abbreviation *backward-terminating*
where *backward-terminating* $x \equiv x \leq -(1;x);x;1$

abbreviation *forward-terminating*
where *forward-terminating* $x \equiv x \leq 1;x;-(x;1)$

abbreviation *terminating*
where *terminating* $x \equiv x \leq -(1;x);x;1 \cdot 1;x;-(x;1)$

abbreviation *backward-finite*
where *backward-finite* $x \equiv x \leq x^{T\star} + -(1;x);x;1$

abbreviation *forward-finite*
where *forward-finite* $x \equiv x \leq x^{T\star} + 1;x;-(x;1)$

abbreviation *finite*
where *finite* $x \equiv x \leq x^{T\star} + (-(1;x);x;1 \cdot 1;x;-(x;1))$

abbreviation *no-start-points-path*
where *no-start-points-path* $x \equiv$ *path* $x \wedge$ *no-start-points* x

abbreviation *no-end-points-path*
where *no-end-points-path* $x \equiv$ *path* $x \wedge$ *no-end-points* x

abbreviation *no-start-end-points-path*
where *no-start-end-points-path* $x \equiv$ *path* $x \wedge$ *no-start-end-points* x

abbreviation *has-start-points-path*
where *has-start-points-path* $x \equiv$ *path* $x \wedge$ *has-start-points* x

abbreviation *has-end-points-path*
where *has-end-points-path* $x \equiv$ *path* $x \wedge$ *has-end-points* x

abbreviation *has-start-end-points-path*
where *has-start-end-points-path* $x \equiv$ *path* $x \wedge$ *has-start-end-points* x

abbreviation *backward-terminating-path*
where *backward-terminating-path* $x \equiv$ *path* $x \wedge$ *backward-terminating* x

abbreviation *forward-terminating-path*
where *forward-terminating-path* $x \equiv$ *path* $x \wedge$ *forward-terminating* x

abbreviation *terminating-path*
where *terminating-path* $x \equiv$ *path* $x \wedge$ *terminating* x

abbreviation *backward-finite-path*
where *backward-finite-path* $x \equiv$ *path* $x \wedge$ *backward-finite* x

abbreviation *forward-finite-path*
where *forward-finite-path* $x \equiv$ *path* $x \wedge$ *forward-finite* x

abbreviation *finite-path*
where *finite-path* $x \equiv$ *path* $x \wedge$ *finite* x

General properties

lemma *reachability-from-z-in-y*:
assumes $x \leq y^\star;z$
and $x \cdot z = 0$
shows $x \leq y^+;z$
⟨*proof*⟩

lemma *reachable-imp*:
assumes *point* p
and *point* q
and $p^\star;q \leq p^{T\star};p$
shows $p \leq p^\star;q$
⟨*proof*⟩

Basic equivalences

lemma *no-start-end-points-iff*:
no-start-end-points $x \longleftrightarrow$ *no-start-points* $x \wedge$ *no-end-points* x
⟨*proof*⟩

lemma *has-start-end-points-iff*:
has-start-end-points $x \longleftrightarrow$ *has-start-points* $x \wedge$ *has-end-points* x
⟨*proof*⟩

lemma *terminating-iff*:
terminating $x \longleftrightarrow$ *backward-terminating* $x \wedge$ *forward-terminating* x
⟨*proof*⟩

lemma *finite-iff*:
finite $x \longleftrightarrow$ *backward-finite* $x \wedge$ *forward-finite* x
⟨*proof*⟩

lemma *no-start-end-points-path-iff*:
no-start-end-points-path $x \longleftrightarrow$ *no-start-points-path* $x \wedge$ *no-end-points-path* x
⟨*proof*⟩

lemma *has-start-end-points-path-iff*:

has-start-end-points-path $x \longleftrightarrow$ *has-start-points-path* $x \wedge$ *has-end-points-path* x
⟨*proof*⟩

lemma *terminating-path-iff*:
terminating-path $x \longleftrightarrow$ *backward-terminating-path* $x \wedge$ *forward-terminating-path* x
⟨*proof*⟩

lemma *finite-path-iff*:
finite-path $x \longleftrightarrow$ *backward-finite-path* $x \wedge$ *forward-finite-path* x
⟨*proof*⟩

Closure under converse

lemma *connected-conv*:
connected $x \longleftrightarrow$ *connected* (x^T)
⟨*proof*⟩

lemma *conv-many-strongly-connected*:
many-strongly-connected $x \longleftrightarrow$ *many-strongly-connected* (x^T)
⟨*proof*⟩

lemma *conv-one-strongly-connected*:
one-strongly-connected $x \longleftrightarrow$ *one-strongly-connected* (x^T)
⟨*proof*⟩

lemma *conv-path*:
path $x \longleftrightarrow$ *path* (x^T)
⟨*proof*⟩

lemma *conv-cycle*:
cycle $x \longleftrightarrow$ *cycle* (x^T)
⟨*proof*⟩

lemma *conv-no-start-points*:
no-start-points $x \longleftrightarrow$ *no-end-points* (x^T)
⟨*proof*⟩

lemma *conv-no-start-end-points*:
no-start-end-points $x \longleftrightarrow$ *no-start-end-points* (x^T)
⟨*proof*⟩

lemma *conv-has-start-points*:
has-start-points $x \longleftrightarrow$ *has-end-points* (x^T)
⟨*proof*⟩

lemma *conv-has-start-end-points*:
has-start-end-points $x \longleftrightarrow$ *has-start-end-points* (x^T)
⟨*proof*⟩

lemma *conv-backward-terminating*:
backward-terminating $x \longleftrightarrow$ *forward-terminating* (x^T)
⟨*proof*⟩

lemma *conv-terminating*:
terminating $x \longleftrightarrow$ *terminating* (x^T)
⟨*proof*⟩

lemma *conv-backward-finite*:
backward-finite $x \longleftrightarrow$ *forward-finite* (x^T)
⟨*proof*⟩

lemma *conv-finite*:
finite $x \longleftrightarrow$ *finite* (x^T)
⟨*proof*⟩

lemma *conv-no-start-points-path*:
no-start-points-path $x \longleftrightarrow$ *no-end-points-path* (x^T)
⟨*proof*⟩

lemma *conv-no-start-end-points-path*:
no-start-end-points-path $x \longleftrightarrow$ *no-start-end-points-path* (x^T)
⟨*proof*⟩

lemma *conv-has-start-points-path*:
has-start-points-path $x \longleftrightarrow$ *has-end-points-path* (x^T)
⟨*proof*⟩

lemma *conv-has-start-end-points-path*:
has-start-end-points-path $x \longleftrightarrow$ *has-start-end-points-path* (x^T)
⟨*proof*⟩

lemma *conv-backward-terminating-path*:
backward-terminating-path $x \longleftrightarrow$ *forward-terminating-path* (x^T)
⟨*proof*⟩

lemma *conv-terminating-path*:
terminating-path $x \longleftrightarrow$ *terminating-path* (x^T)
⟨*proof*⟩

lemma *conv-backward-finite-path*:
backward-finite-path $x \longleftrightarrow$ *forward-finite-path* (x^T)
⟨*proof*⟩

lemma *conv-finite-path*:
finite-path $x \longleftrightarrow$ *finite-path* (x^T)
⟨*proof*⟩

Equivalences for *connected*

lemma *connected-iff2*:
assumes *is-inj* x
and *is-p-fun* x
shows *connected* $x \longleftrightarrow x;1;x^T \leq x^\star + x^{T\star}$
⟨*proof*⟩

lemma *connected-iff3*:
assumes *is-inj* x
and *is-p-fun* x
shows *connected* $x \longleftrightarrow x^T;1;x \leq x^\star + x^{T\star}$
⟨*proof*⟩

lemma *connected-iff4*:
connected $x \longleftrightarrow x^T;1;x^T \leq x^\star + x^{T\star}$
⟨*proof*⟩

lemma *connected-iff5*:
connected $x \longleftrightarrow x^+;1;x^+ \leq x^\star + x^{T\star}$
⟨*proof*⟩

lemma *connected-iff6*:
assumes *is-inj* x
and *is-p-fun* x
shows *connected* $x \longleftrightarrow x^+;1;(x^+)^T \leq x^\star + x^{T\star}$
⟨*proof*⟩

lemma *connected-iff7*:
assumes *is-inj* x
and *is-p-fun* x
shows *connected* $x \longleftrightarrow (x^+)^T;1;x^+ \leq x^\star + x^{T\star}$
⟨*proof*⟩

lemma *connected-iff8*:
connected $x \longleftrightarrow (x^+)^T;1;(x^+)^T \leq x^\star + x^{T\star}$
⟨*proof*⟩

Equivalences and implications for *many-strongly-connected*

lemma *many-strongly-connected-iff-1*:
many-strongly-connected $x \longleftrightarrow x^T \leq x^\star$
⟨*proof*⟩

lemma *many-strongly-connected-iff-2*:
many-strongly-connected $x \longleftrightarrow x^T \leq x^+$
⟨*proof*⟩

lemma *many-strongly-connected-iff-3*:
many-strongly-connected $x \longleftrightarrow x \leq x^{T\star}$
⟨*proof*⟩

lemma *many-strongly-connected-iff-4*:
  many-strongly-connected $x \longleftrightarrow x \leq x^{T+}$
⟨*proof*⟩

lemma *many-strongly-connected-iff-5*:
  many-strongly-connected $x \longleftrightarrow x^{\star};x^{T} \leq x^{+}$
⟨*proof*⟩

lemma *many-strongly-connected-iff-6*:
  many-strongly-connected $x \longleftrightarrow x^{T};x^{\star} \leq x^{+}$
⟨*proof*⟩

lemma *many-strongly-connected-iff-7*:
  many-strongly-connected $x \longleftrightarrow x^{T+} = x^{+}$
⟨*proof*⟩

lemma *many-strongly-connected-iff-5-eq*:
  many-strongly-connected $x \longleftrightarrow x^{\star};x^{T} = x^{+}$
⟨*proof*⟩

lemma *many-strongly-connected-iff-6-eq*:
  many-strongly-connected $x \longleftrightarrow x^{T};x^{\star} = x^{+}$
⟨*proof*⟩

lemma *many-strongly-connected-implies-no-start-end-points*:
  assumes *many-strongly-connected* x
    shows *no-start-end-points* x
⟨*proof*⟩

lemma *many-strongly-connected-implies-8*:
  assumes *many-strongly-connected* x
    shows $x;x^{T} \leq x^{+}$
⟨*proof*⟩

lemma *many-strongly-connected-implies-9*:
  assumes *many-strongly-connected* x
    shows $x^{T};x \leq x^{+}$
⟨*proof*⟩

lemma *many-strongly-connected-implies-10*:
  assumes *many-strongly-connected* x
    shows $x;x^{T};x^{\star} \leq x^{+}$
⟨*proof*⟩

lemma *many-strongly-connected-implies-10-eq*:
  assumes *many-strongly-connected* x
    shows $x;x^{T};x^{\star} = x^{+}$
⟨*proof*⟩

lemma *many-strongly-connected-implies-11*:
assumes *many-strongly-connected* x
shows $x^\star;x^T;x \leq x^+$
⟨*proof*⟩

lemma *many-strongly-connected-implies-11-eq*:
assumes *many-strongly-connected* x
shows $x^\star;x^T;x = x^+$
⟨*proof*⟩

lemma *many-strongly-connected-implies-12*:
assumes *many-strongly-connected* x
shows $x^\star;x;x^T \leq x^+$
⟨*proof*⟩

lemma *many-strongly-connected-implies-12-eq*:
assumes *many-strongly-connected* x
shows $x^\star;x;x^T = x^+$
⟨*proof*⟩

lemma *many-strongly-connected-implies-13*:
assumes *many-strongly-connected* x
shows $x^T;x;x^\star \leq x^+$
⟨*proof*⟩

lemma *many-strongly-connected-implies-13-eq*:
assumes *many-strongly-connected* x
shows $x^T;x;x^\star = x^+$
⟨*proof*⟩

lemma *many-strongly-connected-iff-8*:
assumes *is-p-fun* x
shows *many-strongly-connected* $x \longleftrightarrow x;x^T \leq x^+$
⟨*proof*⟩

lemma *many-strongly-connected-iff-9*:
assumes *is-inj* x
shows *many-strongly-connected* $x \longleftrightarrow x^T;x \leq x^+$
⟨*proof*⟩

lemma *many-strongly-connected-iff-10*:
assumes *is-p-fun* x
shows *many-strongly-connected* $x \longleftrightarrow x;x^T;x^\star \leq x^+$
⟨*proof*⟩

lemma *many-strongly-connected-iff-10-eq*:
assumes *is-p-fun* x
shows *many-strongly-connected* $x \longleftrightarrow x;x^T;x^\star = x^+$
⟨*proof*⟩

lemma *many-strongly-connected-iff-11*:
assumes *is-inj* x
shows *many-strongly-connected* $x \longleftrightarrow x^\star;x^T;x \leq x^+$
⟨*proof*⟩

lemma *many-strongly-connected-iff-11-eq*:
assumes *is-inj* x
shows *many-strongly-connected* $x \longleftrightarrow x^\star;x^T;x = x^+$
⟨*proof*⟩

lemma *many-strongly-connected-iff-12*:
assumes *is-p-fun* x
shows *many-strongly-connected* $x \longleftrightarrow x^\star;x;x^T \leq x^+$
⟨*proof*⟩

lemma *many-strongly-connected-iff-12-eq*:
assumes *is-p-fun* x
shows *many-strongly-connected* $x \longleftrightarrow x^\star;x;x^T = x^+$
⟨*proof*⟩

lemma *many-strongly-connected-iff-13*:
assumes *is-inj* x
shows *many-strongly-connected* $x \longleftrightarrow x^T;x;x^\star \leq x^+$
⟨*proof*⟩

lemma *many-strongly-connected-iff-13-eq*:
assumes *is-inj* x
shows *many-strongly-connected* $x \longleftrightarrow x^T;x;x^\star = x^+$
⟨*proof*⟩

Equivalences and implications for *one-strongly-connected*

lemma *one-strongly-connected-iff*:
one-strongly-connected $x \longleftrightarrow$ *connected* $x \wedge$ *many-strongly-connected* x
⟨*proof*⟩

lemma *one-strongly-connected-iff-1*:
one-strongly-connected $x \longleftrightarrow x^T;1;x^T \leq x^+$
⟨*proof*⟩

lemma *one-strongly-connected-iff-1-eq*:
one-strongly-connected $x \longleftrightarrow x^T;1;x^T = x^+$
⟨*proof*⟩

lemma *one-strongly-connected-iff-2*:
one-strongly-connected $x \longleftrightarrow x;1;x \leq x^{T\star}$
⟨*proof*⟩

lemma *one-strongly-connected-iff-3*:

one-strongly-connected $x \longleftrightarrow x;1;x \leq x^{T+}$
⟨*proof*⟩

lemma *one-strongly-connected-iff-3-eq*:
one-strongly-connected $x \longleftrightarrow x;1;x = x^{T+}$
⟨*proof*⟩

lemma *one-strongly-connected-iff-4-eq*:
one-strongly-connected $x \longleftrightarrow x^T;1;x = x^+$
⟨*proof*⟩

lemma *one-strongly-connected-iff-5-eq*:
one-strongly-connected $x \longleftrightarrow x;1;x^T = x^+$
⟨*proof*⟩

lemma *one-strongly-connected-iff-6-aux*:
$x;x^+ \leq x;1;x$
⟨*proof*⟩

lemma *one-strongly-connected-implies-6-eq*:
assumes *one-strongly-connected x*
shows $x;1;x = x;x^+$
⟨*proof*⟩

lemma *one-strongly-connected-iff-7-aux*:
$x^+ \leq x;1;x$
⟨*proof*⟩

lemma *one-strongly-connected-implies-7-eq*:
assumes *one-strongly-connected x*
shows $x;1;x = x^+$
⟨*proof*⟩

lemma *one-strongly-connected-implies-8*:
assumes *one-strongly-connected x*
shows $x;1;x \leq x^\star$
⟨*proof*⟩

lemma *one-strongly-connected-iff-4*:
assumes *is-inj x*
shows *one-strongly-connected* $x \longleftrightarrow x^T;1;x \leq x^+$
⟨*proof*⟩

lemma *one-strongly-connected-iff-5*:
assumes *is-p-fun x*
shows *one-strongly-connected* $x \longleftrightarrow x;1;x^T \leq x^+$
⟨*proof*⟩

lemma *one-strongly-connected-iff-6*:

assumes *is-p-fun* x
and *is-inj* x
shows *one-strongly-connected* $x \longleftrightarrow x;1;x \le x;x^+$
⟨*proof*⟩

lemma *one-strongly-connected-iff-6-eq*:
assumes *is-p-fun* x
and *is-inj* x
shows *one-strongly-connected* $x \longleftrightarrow x;1;x = x;x^+$
⟨*proof*⟩

Start points and end points

lemma *start-end-implies-terminating*:
assumes *has-start-points* x
and *has-end-points* x
shows *terminating* x
⟨*proof*⟩

lemma *start-points-end-points-conv*:
start-points x = *end-points* (x^T)
⟨*proof*⟩

lemma *start-point-at-most-one*:
assumes *path* x
shows *is-inj* (*start-points* x)
⟨*proof*⟩

lemma *start-point-zero-point*:
assumes *path* x
shows *start-points* $x = 0 \vee$ *is-point* (*start-points* x)
⟨*proof*⟩

lemma *start-point-iff1*:
assumes *path* x
shows *is-point* (*start-points* x) $\longleftrightarrow \neg$(*no-start-points* x)
⟨*proof*⟩

lemma *end-point-at-most-one*:
assumes *path* x
shows *is-inj* (*end-points* x)
⟨*proof*⟩

lemma *end-point-zero-point*:
assumes *path* x
shows *end-points* $x = 0 \vee$ *is-point* (*end-points* x)
⟨*proof*⟩

lemma *end-point-iff1*:
assumes *path* x

shows *is-point* (*end-points* x) $\longleftrightarrow$ $\neg$(*no-end-points* x)
⟨*proof*⟩

lemma *predecessor-point'*:
assumes *path* x
and *point* s
and *point* e
and $e;s^T \leq x$
shows $x;s = e$
⟨*proof*⟩

lemma *predecessor-point*:
assumes *path* x
and *point* s
and *point* e
and $e;s^T \leq x$
shows *point*$(x;s)$
⟨*proof*⟩

lemma *points-of-path-iff*:
shows $(x + x^T);1 = x^T;1 +$ *start-points*(x)
and $(x + x^T);1 = x;1 +$ *end-points*(x)
⟨*proof*⟩

Path concatenation preliminaries

lemma *path-concat-aux-1*:
assumes $x;1 \cdot y;1 \cdot y^T;1 = 0$
and *end-points* x = *start-points* y
shows $x;1 \cdot y;1 = 0$
⟨*proof*⟩

lemma *path-concat-aux-2*:
assumes $x;1 \cdot x^T;1 \cdot y^T;1 = 0$
and *end-points* x = *start-points* y
shows $x^T;1 \cdot y^T;1 = 0$
⟨*proof*⟩

lemma *path-concat-aux3-1*:
assumes *path* x
shows $x;1;x^T \leq x^\star + x^{T\star}$
⟨*proof*⟩

lemma *path-concat-aux3-2*:
assumes *path* x
shows $x^T;1;x \leq x^\star + x^{T\star}$
⟨*proof*⟩

lemma *path-concat-aux3-3*:
assumes *path* x

shows $x^T;1;x^T \le x^\star + x^{T\star}$
⟨*proof*⟩

lemma *path-concat-aux-3*:
assumes *path x*
and $y \le x^+ + x^{T+}$
and $z \le x^+ + x^{T+}$
shows $y;1;z \le x^\star + x^{T\star}$
⟨*proof*⟩

lemma *path-concat-aux-4*:
$x^\star + x^{T\star} \le x^\star + x^T;1$
⟨*proof*⟩

lemma *path-concat-aux-5*:
assumes *path x*
and $y \le$ *start-points x*
and $z \le x + x^T$
shows $y;1;z \le x^\star$
⟨*proof*⟩

lemma *path-conditions-disjoint-points-iff*:
$x;1 \cdot (x^T;1 + y;1) \cdot y^T;1 = 0 \wedge$ *start-points x* $\cdot$ *end-points y* $= 0 \longleftrightarrow x;1 \cdot y^T;1 = 0$
⟨*proof*⟩

end

2.2 Consequences with the Tarski rule

context *relation-algebra-rtc-tarski*
begin

General theorems

lemma *reachable-implies-predecessor*:
assumes $p \ne q$
and *point p*
and *point q*
and $x^\star;q \le x^{T\star};p$
shows $x;q \ne 0$
⟨*proof*⟩

lemma *acyclic-imp-one-step-different-points*:
assumes *is-acyclic x*
and *point p*
and *point q*
and $p \le x;q$
shows $p \le -q$ **and** $p \ne q$
⟨*proof*⟩

Start points and end points

lemma *start-point-iff2*:
assumes *path* x
shows *is-point* (*start-points* x) $\longleftrightarrow$ *has-start-points* x
⟨*proof*⟩

lemma *end-point-iff2*:
assumes *path* x
shows *is-point* (*end-points* x) $\longleftrightarrow$ *has-end-points* x
⟨*proof*⟩

lemma *edge-is-path*:
assumes *is-point* p
and *is-point* q
shows *path* $(p;q^T)$
⟨*proof*⟩

lemma *edge-start*:
assumes *is-point* p
and *is-point* q
and $p \neq q$
shows *start-points* $(p;q^T) = p$
⟨*proof*⟩

lemma *edge-end*:
assumes *is-point* p
and *is-point* q
and $p \neq q$
shows *end-points* $(p;q^T) = q$
⟨*proof*⟩

lemma *loop-no-start*:
assumes *is-point* p
shows *start-points* $(p;p^T) = 0$
⟨*proof*⟩

lemma *loop-no-end*:
assumes *is-point* p
shows *end-points* $(p;p^T) = 0$
⟨*proof*⟩

lemma *start-point-no-predecessor*:
x;*start-points*$(x) = 0$
⟨*proof*⟩

lemma *end-point-no-successor*:
x^T;*end-points*$(x) = 0$
⟨*proof*⟩

lemma *start-to-end*:
assumes *path* x
shows *start-points*(x);*end-points*$(x)^T \leq x^\star$
⟨*proof*⟩

lemma *path-acyclic*:
assumes *has-start-points-path* x
shows *is-acyclic* x
⟨*proof*⟩

Equivalences for *terminating*

lemma *backward-terminating-iff1*:
assumes *path* x
shows *backward-terminating* $x \longleftrightarrow$ *has-start-points* $x \vee x = 0$
⟨*proof*⟩

lemma *backward-terminating-iff2-aux*:
assumes *path* x
shows $x;1 \cdot 1;x^T \cdot -(1;x) \leq x^{T\star}$
⟨*proof*⟩

lemma *backward-terminating-iff2*:
assumes *path* x
shows *backward-terminating* $x \longleftrightarrow x \leq x^{T\star};-(x^T;1)$
⟨*proof*⟩

lemma *backward-terminating-iff3-aux*:
assumes *path* x
shows $x^T;1 \cdot 1;x^T \cdot -(1;x) \leq x^{T\star}$
⟨*proof*⟩

lemma *backward-terminating-iff3*:
assumes *path* x
shows *backward-terminating* $x \longleftrightarrow x^T \leq x^{T\star};-(x^T;1)$
⟨*proof*⟩

lemma *backward-terminating-iff4*:
assumes *path* x
shows *backward-terminating* $x \longleftrightarrow x \leq -(1;x);x^\star$
⟨*proof*⟩

lemma *forward-terminating-iff1*:
assumes *path* x
shows *forward-terminating* $x \longleftrightarrow$ *has-end-points* $x \vee x = 0$
⟨*proof*⟩

lemma *forward-terminating-iff2*:
assumes *path* x
shows *forward-terminating* $x \longleftrightarrow x^T \leq x^\star;-(x;1)$

⟨*proof*⟩

lemma *forward-terminating-iff3*:
  assumes *path x*
    shows *forward-terminating x* $\longleftrightarrow x \leq x^\star;-(x;1)$
⟨*proof*⟩

lemma *forward-terminating-iff4*:
  assumes *path x*
    shows *forward-terminating x* $\longleftrightarrow x \leq -(1;x^T);x^{T\star}$
⟨*proof*⟩

lemma *terminating-iff1*:
  assumes *path x*
    shows *terminating x* $\longleftrightarrow$ *has-start-end-points x* $\vee x = 0$
⟨*proof*⟩

lemma *terminating-iff2*:
  assumes *path x*
    shows *terminating x* $\longleftrightarrow x \leq x^{T\star};-(x^T;1) \cdot -(1;x^T);x^{T\star}$
⟨*proof*⟩

lemma *terminating-iff3*:
  assumes *path x*
    shows *terminating x* $\longleftrightarrow x \leq x^\star;-(x;1) \cdot -(1;x);x^\star$
⟨*proof*⟩

lemma *backward-terminating-path-irreflexive*:
  assumes *backward-terminating-path x*
    shows $x \leq -1'$
⟨*proof*⟩

lemma *forward-terminating-path-end-points-1*:
  assumes *forward-terminating-path x*
    shows $x \leq x^+$;*end-points x*
⟨*proof*⟩

lemma *forward-terminating-path-end-points-2*:
  assumes *forward-terminating-path x*
    shows $x^T \leq x^\star$;*end-points x*
⟨*proof*⟩

lemma *forward-terminating-path-end-points-3*:
  assumes *forward-terminating-path x*
  shows *start-points x* $\leq x^+$;*end-points x*
⟨*proof*⟩

lemma *backward-terminating-path-start-points-1*:
  assumes *backward-terminating-path x*

shows $x^T \le x^{T+}$;*start-points* x
⟨*proof*⟩

lemma *backward-terminating-path-start-points-2*:
assumes *backward-terminating-path* x
shows $x \le x^{T\star}$;*start-points* x
⟨*proof*⟩

lemma *backward-terminating-path-start-points-3*:
assumes *backward-terminating-path* x
shows *end-points* $x \le x^{T+}$;*start-points* x
⟨*proof*⟩

lemma *path-aux1a*:
assumes *forward-terminating-path* x
shows $x \ne 0 \longleftrightarrow$ *end-points* $x \ne 0$
⟨*proof*⟩

lemma *path-aux1b*:
assumes *backward-terminating-path* y
shows $y \ne 0 \longleftrightarrow$ *start-points* $y \ne 0$
⟨*proof*⟩

lemma *path-aux1*:
assumes *forward-terminating-path* x
and *backward-terminating-path* y
shows $x \ne 0 \vee y \ne 0 \longleftrightarrow$ *end-points* $x \ne 0 \vee$ *start-points* $y \ne 0$
⟨*proof*⟩

Equivalences for *finite*

lemma *backward-finite-iff-msc*:
backward-finite $x \longleftrightarrow$ *many-strongly-connected* $x \vee$ *backward-terminating* x
⟨*proof*⟩

lemma *forward-finite-iff-msc*:
forward-finite $x \longleftrightarrow$ *many-strongly-connected* $x \vee$ *forward-terminating* x
⟨*proof*⟩

lemma *finite-iff-msc*:
finite $x \longleftrightarrow$ *many-strongly-connected* $x \vee$ *terminating* x
⟨*proof*⟩

Path concatenation

lemma *path-concatenation*:
assumes *forward-terminating-path* x
and *backward-terminating-path* y

and *end-points* x = *start-points* y
and $x;1 \cdot (x^T;1 + y;1) \cdot y^T;1 = 0$
shows *path* $(x+y)$
⟨*proof*⟩

lemma *path-concatenation-with-edge*:
assumes $x \neq 0$
and *forward-terminating-path* x
and *is-point* q
and $q \leq -(1;x)$
shows *path* $(x+($*end-points* $x);q^T)$
⟨*proof*⟩

lemma *path-concatenation-cycle-free*:
assumes *forward-terminating-path* x
and *backward-terminating-path* y
and *end-points* x = *start-points* y
and $x;1 \cdot y^T;1 = 0$
shows *path* $(x+y)$
⟨*proof*⟩

lemma *path-concatenation-start-points-approx*:
assumes *end-points* x = *start-points* y
shows *start-points* $(x+y) \leq$ *start-points* x
⟨*proof*⟩

lemma *path-concatenation-end-points-approx*:
assumes *end-points* x = *start-points* y
shows *end-points* $(x+y) \leq$ *end-points* y
⟨*proof*⟩

lemma *path-concatenation-start-points*:
assumes *end-points* x = *start-points* y
and $x;1 \cdot y^T;1 = 0$
shows *start-points* $(x+y)$ = *start-points* x
⟨*proof*⟩

lemma *path-concatenation-end-points*:
assumes *end-points* x = *start-points* y
and $x;1 \cdot y^T;1 = 0$
shows *end-points* $(x+y)$ = *end-points* y
⟨*proof*⟩

lemma *path-concatenation-cycle-free-complete*:
assumes *forward-terminating-path* x
and *backward-terminating-path* y
and *end-points* x = *start-points* y
and $x;1 \cdot y^T;1 = 0$
shows *path* $(x+y) \wedge$ *start-points* $(x+y)$ = *start-points* $x \wedge$ *end-points* $(x+y)$

$=$ *end-points* y
⟨*proof*⟩

Path restriction (path from a given point)

lemma *reachable-points-iff*:
assumes *point* p
shows $(x^{T\star};p \cdot x) = (x^{T\star};p \cdot 1');x$
⟨*proof*⟩

lemma *path-from-given-point*:
assumes *path* x
and *point* p
shows $path(x^{T\star};p \cdot x)$
and $start\text{-}points(x^{T\star};p \cdot x) \leq p$
and $end\text{-}points(x^{T\star};p \cdot x) \leq end\text{-}points(x)$
⟨*proof*⟩

lemma *path-from-given-point′*:
assumes *has-start-points-path* x
and *point* p
and $p \leq x;1$
shows $path(x^{T\star};p \cdot x)$
and $start\text{-}points(x^{T\star};p \cdot x) = p$
and $end\text{-}points(x^{T\star};p \cdot x) = end\text{-}points(x)$
⟨*proof*⟩

Cycles

lemma *selfloop-is-cycle*:
assumes *is-point* x
shows *cycle* $(x;x^T)$
⟨*proof*⟩

lemma *start-point-no-cycle*:
assumes *has-start-points-path* x
shows $\neg$ *cycle* x
⟨*proof*⟩

lemma *end-point-no-cycle*:
assumes *has-end-points-path* x
shows $\neg$ *cycle* x
⟨*proof*⟩

lemma *cycle-no-points*:
assumes *cycle* x
shows *start-points* $x = 0$
and *end-points* $x = 0$
⟨*proof*⟩

Path concatenation to cycle

lemma *path-path-equals-cycle-aux*:
  assumes *has-start-end-points-path x*
    and *has-start-end-points-path y*
    and *start-points x = end-points y*
    and *end-points x = start-points y*
shows $x \leq (x+y)^{T\star}$
⟨*proof*⟩

lemma *path-path-equals-cycle*:
  assumes *has-start-end-points-path x*
    and *has-start-end-points-path y*
    and *start-points x = end-points y*
    and *end-points x = start-points y*
    and $x;1 \cdot (x^T;1 + y;1) \cdot y^T;1 = 0$
  shows *cycle*$(x + y)$
⟨*proof*⟩

lemma *path-edge-equals-cycle*:
  assumes *has-start-end-points-path x*
  shows *cycle*$(x +$ *end-points*(x);$($*start-points* $x)^T)$
⟨*proof*⟩

Break cycles

lemma *cycle-remove-edge*:
  assumes *cycle x*
    and *point s*
    and *point e*
    and $e;s^T \leq x$
  shows *path*$(x \cdot -(e;s^T))$
    and *start-points* $(x \cdot -(e;s^T)) \leq s$
    and *end-points* $(x \cdot -(e;s^T)) \leq e$
⟨*proof*⟩

lemma *cycle-remove-edge'*:
  assumes *cycle x*
    and *point s*
    and *point e*
    and $s \neq e$
    and $e;s^T \leq x$
  shows *path*$(x \cdot -(e;s^T))$
    and $s =$ *start-points* $(x \cdot -(e;s^T))$
    and $e =$ *end-points* $(x \cdot -(e;s^T))$
⟨*proof*⟩

end

end

3 Relational Characterisation of Rooted Paths

We characterise paths together with a designated root. This is important as often algorithms start with a single vertex, and then build up a path, a tree or another structure. An example is Dijkstra's shortest path algorithm.

theory *Rooted-Paths*

imports *Paths*

begin

context *relation-algebra*
begin

General theorems

lemma *step-has-target*:
assumes $x;r \neq 0$
shows $x^T;1 \neq 0$
⟨*proof*⟩

lemma *end-point-char*:
$x^T;p = 0 \longleftrightarrow p \leq -(x;1)$
⟨*proof*⟩

end

context *relation-algebra-tarski*
begin

General theorems concerning points

lemma *successor-point*:
assumes *is-inj* x
and *point* r
and $x;r \neq 0$
shows *point* $(x;r)$
⟨*proof*⟩

lemma *no-end-point-char*:
assumes *point* p
shows $x^T;p \neq 0 \longleftrightarrow p \leq x;1$
⟨*proof*⟩

lemma *no-end-point-char-converse*:
assumes *point* p
shows $x;p \neq 0 \longleftrightarrow p \leq x^T;1$
⟨*proof*⟩

end

3.1 Consequences without the Tarski rule

context *relation-algebra-rtc*
begin

Definitions for path classifications

definition *path-root*
where *path-root* r x $\equiv$ $r;x \leq x^\star + x^{T\star} \wedge$ *is-inj* $x \wedge$ *is-p-fun* $x \wedge$ *point* r

abbreviation *connected-root*
where *connected-root* r x $\equiv$ $r;x \leq x^+$

definition *backward-finite-path-root*
where *backward-finite-path-root* r x $\equiv$ *connected-root* r x $\wedge$ *is-inj* x $\wedge$ *is-p-fun* x $\wedge$ *point* r

abbreviation *backward-terminating-path-root*
where *backward-terminating-path-root* r x $\equiv$ *backward-finite-path-root* r x $\wedge$ $x;r = 0$

abbreviation *cycle-root*
where *cycle-root* r x $\equiv$ $r;x \leq x^+ \cdot x^T;1 \wedge$ *is-inj* $x \wedge$ *is-p-fun* $x \wedge$ *point* r

abbreviation *non-empty-cycle-root*
where *non-empty-cycle-root* r x $\equiv$ *backward-finite-path-root* r x $\wedge$ $r \leq x^T;1$

abbreviation *finite-path-root-end*
where *finite-path-root-end* r x e $\equiv$ *backward-finite-path-root* r x $\wedge$ *point* e $\wedge$ $r \leq x^\star;e$

abbreviation *terminating-path-root-end*
where *terminating-path-root-end* r x e $\equiv$ *finite-path-root-end* r x e $\wedge$ $x^T;e = 0$

Equivalent formulations of *connected-root*

lemma *connected-root-iff1*:
assumes *point* r
shows *connected-root* r x $\longleftrightarrow$ $1;x \leq r^T;x^+$
⟨*proof*⟩

lemma *connected-root-iff2*:
assumes *point* r
shows *connected-root* r x $\longleftrightarrow$ $x^T;1 \leq x^{T+};r$
⟨*proof*⟩

lemma *connected-root-aux*:
$x^{T+};r \leq x^T;1$
⟨*proof*⟩

lemma *connected-root-iff3*:
assumes *point* r

shows *connected-root r x* $\longleftrightarrow$ $x^T;1 = x^{T+};r$
⟨*proof*⟩

lemma *connected-root-iff4*:
assumes *point r*
shows *connected-root r x* $\longleftrightarrow$ $1;x = r^T;x^+$
⟨*proof*⟩

Consequences of *connected-root*

lemma *has-root-contra*:
assumes *connected-root r x*
and *point r*
and $x^T;r = 0$
shows $x = 0$
⟨*proof*⟩

lemma *has-root*:
assumes *connected-root r x*
and *point r*
and $x \neq 0$
shows $x^T;r \neq 0$
⟨*proof*⟩

lemma *connected-root-move-root*:
assumes *connected-root r x*
and $q \leq x^\star;r$
shows *connected-root q x*
⟨*proof*⟩

lemma *root-cycle-converse*:
assumes *connected-root r x*
and *point r*
and $x;r \neq 0$
shows $x^T;r \neq 0$
⟨*proof*⟩

Rooted paths

lemma *path-iff-aux-1*:
assumes *bijective r*
shows $r;x \leq x^\star + x^{T\star} \longleftrightarrow x \leq r^T;(x^\star + x^{T\star})$
⟨*proof*⟩

lemma *path-iff-aux-2*:
assumes *bijective r*
shows $r;x \leq x^\star + x^{T\star} \longleftrightarrow x^T \leq (x^\star + x^{T\star});r$
⟨*proof*⟩

lemma *path-iff-backward*:
assumes *is-inj x*

and *is-p-fun* x
and *point* r
and $r;x \leq x^\star + x^{T\star}$
shows *connected* x
⟨*proof*⟩

lemma *empty-path-root-end*:
assumes *terminating-path-root-end* $r\ x\ e$
shows $e = r \longleftrightarrow x = 0$
⟨*proof*⟩

lemma *path-root-acyclic*:
assumes *path-root* $r\ x$
and $x;r = 0$
shows *is-acyclic* x
⟨*proof*⟩

Start points and end points

lemma *start-points-in-root-aux*:
assumes *backward-finite-path-root* $r\ x$
shows $x;1 \leq x^{T\star};r$
⟨*proof*⟩

lemma *start-points-in-root*:
assumes *backward-finite-path-root* $r\ x$
shows *start-points* $x \leq r$
⟨*proof*⟩

lemma *start-points-not-zero-contra*:
assumes *connected-root* $r\ x$
and *point* r
and *start-points* $x = 0$
and $x;r = 0$
shows $x = 0$
⟨*proof*⟩

lemma *start-points-not-zero*:
assumes *connected-root* $r\ x$
and *point* r
and $x \neq 0$
and $x;r = 0$
shows *start-points* $x \neq 0$
⟨*proof*⟩

Backwards terminating and backwards finite

lemma *backward-terminating-path-root-aux*:
assumes *backward-terminating-path-root* $r\ x$
shows $x \leq x^{T\star};-(x^T;1)$
⟨*proof*⟩

lemma *backward-finite-path-connected-aux*:
  assumes *backward-finite-path-root r x*
    shows $x^T;r;x^T \leq x^\star + x^{T\star}$
⟨*proof*⟩

lemma *backward-finite-path-connected*:
  assumes *backward-finite-path-root r x*
    shows *connected x*
⟨*proof*⟩

lemma *backward-finite-path-root-path*:
  assumes *backward-finite-path-root r x*
    shows *path x*
⟨*proof*⟩

lemma *backward-finite-path-root-path-root*:
  assumes *backward-finite-path-root r x*
    shows *path-root r x*
⟨*proof*⟩

lemma *zero-backward-terminating-path-root*:
  assumes *point r*
    shows *backward-terminating-path-root r 0*
⟨*proof*⟩

lemma *backward-finite-path-root-move-root*:
  assumes *backward-finite-path-root r x*
      and *point q*
      and $q \leq x^\star;r$
    shows *backward-finite-path-root q x*
⟨*proof*⟩

Cycle

lemma *non-empty-cycle-root-var-axioms-1*:
  non-empty-cycle-root r x $\longleftrightarrow x^T;1 \leq x^{T+};r \wedge$ *is-inj x* $\wedge$ *is-p-fun x* $\wedge$ *point r* $\wedge$ $r \leq x^T;1$
⟨*proof*⟩

lemma *non-empty-cycle-root-loop*:
  assumes *non-empty-cycle-root r x*
    shows $r \leq x^{T+};r$
⟨*proof*⟩

lemma *cycle-root-end-empty*:
  assumes *terminating-path-root-end r x e*
      and *many-strongly-connected x*
    shows $x = 0$
⟨*proof*⟩

lemma *cycle-root-end-empty-var*:
assumes *terminating-path-root-end r x e*
and $x \neq 0$
shows $\neg$ *many-strongly-connected x*
⟨*proof*⟩

Terminating path

lemma *terminating-path-root-end-connected*:
assumes *terminating-path-root-end r x e*
shows $x;1 \leq x^{+};e$
⟨*proof*⟩

lemma *terminating-path-root-end-forward-finite*:
assumes *terminating-path-root-end r x e*
shows *backward-finite-path-root e* (x^{T})
⟨*proof*⟩

end

3.2 Consequences with the Tarski rule

context *relation-algebra-rtc-tarski*
begin

Some (more) results about points

lemma *point-reachable-converse*:
assumes *is-vector v*
and $v \neq 0$
and *point r*
and $v \leq x^{T+};r$
shows $r \leq x^{+};v$
⟨*proof*⟩

Roots

lemma *root-in-start-points*:
assumes *connected-root r x*
and *is-vector r*
and $x \neq 0$
and $x;r = 0$
shows $r \leq$ *start-points x*
⟨*proof*⟩

lemma *root-equals-start-points*:
assumes *backward-terminating-path-root r x*
and $x \neq 0$
shows $r =$ *start-points x*
⟨*proof*⟩

lemma *root-equals-end-points*:
assumes *backward-terminating-path-root* r (x^T)
and $x \neq 0$
shows $r = \mathit{end\text{-}points}\ x$
⟨*proof*⟩

lemma *root-in-edge-sources*:
assumes *connected-root* r x
and $x \neq 0$
and *is-vector* r
shows $r \leq x;1$
⟨*proof*⟩

Rooted Paths

lemma *non-empty-path-root-iff-aux*:
assumes *path-root* r x
and $x \neq 0$
shows $r \leq (x + x^T);1$
⟨*proof*⟩

Backwards terminating and backwards finite

lemma *backward-terminating-path-root-2*:
assumes *backward-terminating-path-root* r x
shows *backward-terminating* x
⟨*proof*⟩

lemma *backward-terminating-path-root*:
assumes *backward-terminating-path-root* r x
shows *backward-terminating-path* x
⟨*proof*⟩

(Non-empty) Cycle

lemma *cycle-iff*:
assumes *point* r
shows $x;r \neq 0 \longleftrightarrow r \leq x^T;1$
⟨*proof*⟩

lemma *non-empty-cycle-root-iff*:
assumes *connected-root* r x
and *point* r
shows $x;r \neq 0 \longleftrightarrow r \leq x^{T+};r$
⟨*proof*⟩

lemma *backward-finite-path-root-terminating-or-cycle*:
backward-finite-path-root r x $\longleftrightarrow$ *backward-terminating-path-root* r x $\vee$ *non-empty-cycle-root* r x
⟨*proof*⟩

lemma *non-empty-cycle-root-msc*:

assumes *non-empty-cycle-root r x*
shows *many-strongly-connected x*
⟨*proof*⟩

lemma *non-empty-cycle-root-msc-cycle*:
assumes *non-empty-cycle-root r x*
shows *cycle x*
⟨*proof*⟩

lemma *non-empty-cycle-root-non-empty*:
assumes *non-empty-cycle-root r x*
shows $x \neq 0$
⟨*proof*⟩

lemma *non-empty-cycle-root-rtc-symmetric*:
assumes *non-empty-cycle-root r x*
shows $x^\star;r = x^{T\star};r$
⟨*proof*⟩

lemma *non-empty-cycle-root-point-exchange*:
assumes *non-empty-cycle-root r x*
and *point p*
shows $r \leq x^\star;p \longleftrightarrow p \leq x^\star;r$
⟨*proof*⟩

lemma *non-empty-cycle-root-rtc-tc*:
assumes *non-empty-cycle-root r x*
shows $x^\star;r = x^+;r$
⟨*proof*⟩

lemma *non-empty-cycle-root-no-start-end-points*:
assumes *non-empty-cycle-root r x*
shows $x;1 = x^T;1$
⟨*proof*⟩

lemma *non-empty-cycle-root-move-root*:
assumes *non-empty-cycle-root r x*
and *point q*
and $q \leq x^\star;r$
shows *non-empty-cycle-root q x*
⟨*proof*⟩

lemma *non-empty-cycle-root-loop-converse*:
assumes *non-empty-cycle-root r x*
shows $r \leq x^+;r$
⟨*proof*⟩

lemma *non-empty-cycle-root-move-root-same-reachable*:
assumes *non-empty-cycle-root r x*

and *point* q
and $q \leq x^\star;r$
shows $x^\star;r = x^\star;q$
⟨*proof*⟩

lemma *non-empty-cycle-root-move-root-same-reachable-2*:
assumes *non-empty-cycle-root* r x
and *point* q
and $q \leq x^\star;r$
shows $x^\star;r = x^{T\star};q$
⟨*proof*⟩

lemma *non-empty-cycle-root-move-root-msc*:
assumes *non-empty-cycle-root* r x
shows $x^{T\star};q = x^\star;q$
⟨*proof*⟩

lemma *non-empty-cycle-root-move-root-rtc-tc*:
assumes *non-empty-cycle-root* r x
and *point* q
and $q \leq x^\star;r$
shows $x^\star;q = x^+;q$
⟨*proof*⟩

lemma *non-empty-cycle-root-move-root-loop-converse*:
assumes *non-empty-cycle-root* r x
and *point* q
and $q \leq x^\star;r$
shows $q \leq x^{T+};q$
⟨*proof*⟩

lemma *non-empty-cycle-root-move-root-loop*:
assumes *non-empty-cycle-root* r x
and *point* q
and $q \leq x^\star;r$
shows $q \leq x^+;q$
⟨*proof*⟩

lemma *non-empty-cycle-root-msc-plus*:
assumes *non-empty-cycle-root* r x
shows $x^+;r = x^{T+};r$
⟨*proof*⟩

lemma *non-empty-cycle-root-tc-start-points*:
assumes *non-empty-cycle-root* r x
shows $x^+;r = x;1$
⟨*proof*⟩

lemma *non-empty-cycle-root-rtc-start-points*:

assumes *non-empty-cycle-root r x*
shows $x^\star;r = x;1$
⟨*proof*⟩

lemma *non-empty-cycle-root-converse-start-end-points*:
assumes *non-empty-cycle-root r x*
shows $x^T \leq x;1;x$
⟨*proof*⟩

lemma *non-empty-cycle-root-start-end-points-plus*:
assumes *non-empty-cycle-root r x*
shows $x;1;x \leq x^+$
⟨*proof*⟩

lemma *non-empty-cycle-root-converse-plus*:
assumes *non-empty-cycle-root r x*
shows $x^T \leq x^+$
⟨*proof*⟩

lemma *non-empty-cycle-root-plus-converse*:
assumes *non-empty-cycle-root r x*
shows $x^+ = x^{T+}$
⟨*proof*⟩

lemma *non-empty-cycle-root-converse*:
assumes *non-empty-cycle-root r x*
shows *non-empty-cycle-root r* (x^T)
⟨*proof*⟩

lemma *non-empty-cycle-root-move-root-forward*:
assumes *non-empty-cycle-root r x*
and *point q*
and $r \leq x^\star;q$
shows *non-empty-cycle-root q x*
⟨*proof*⟩

lemma *non-empty-cycle-root-move-root-forward-cycle*:
assumes *non-empty-cycle-root r x*
and *point q*
and $r \leq x^\star;q$
shows $x;q \neq 0 \wedge x^T;q \neq 0$
⟨*proof*⟩

lemma *non-empty-cycle-root-equivalences*:
assumes *non-empty-cycle-root r x*
and *point q*
shows $(r \leq x^\star;q \longleftrightarrow q \leq x^\star;r)$
and $(r \leq x^\star;q \longleftrightarrow x;q \neq 0)$
and $(r \leq x^\star;q \longleftrightarrow x^T;q \neq 0)$

and $(r \leq x^\star;q \longleftrightarrow q \leq x;1)$
and $(r \leq x^\star;q \longleftrightarrow q \leq x^T;1)$
⟨*proof*⟩

lemma *non-empty-cycle-root-chord*:
assumes *non-empty-cycle-root r x*
and *point p*
and *point q*
and $r \leq x^\star;p$
and $r \leq x^\star;q$
shows $p \leq x^\star;q$
⟨*proof*⟩

lemma *non-empty-cycle-root-var-axioms-2*:
non-empty-cycle-root r x $\longleftrightarrow x;1 \leq x^+;r \wedge$ *is-inj x* $\wedge$ *is-p-fun x* $\wedge$ *point r* $\wedge\ r \leq x;1$
⟨*proof*⟩

lemma *non-empty-cycle-root-var-axioms-3*:
non-empty-cycle-root r x $\longleftrightarrow x;1 \leq x^+;r \wedge$ *is-inj x* $\wedge$ *is-p-fun x* $\wedge$ *point r* $\wedge\ r \leq x^+;x;1$
⟨*proof*⟩

lemma *non-empty-cycle-root-subset-equals*:
assumes *non-empty-cycle-root r x*
and *non-empty-cycle-root r y*
and $x \leq y$
shows $x = y$
⟨*proof*⟩

lemma *non-empty-cycle-root-subset-equals-change-root*:
assumes *non-empty-cycle-root r x*
and *non-empty-cycle-root q y*
and $x \leq y$
shows $x = y$
⟨*proof*⟩

lemma *non-empty-cycle-root-equivalences-2*:
assumes *non-empty-cycle-root r x*
shows $(v \leq x^\star;r \longleftrightarrow v \leq x^T;1)$
and $(v \leq x^\star;r \longleftrightarrow v \leq x;1)$
⟨*proof*⟩

lemma *cycle-root-non-empty*:
assumes $x \neq 0$
shows *cycle-root r x* $\longleftrightarrow$ *non-empty-cycle-root r x*
⟨*proof*⟩

Start points and end points

lemma *start-points-path-aux*:
assumes *backward-finite-path-root r x*
and *start-points x* $\neq$ *0*
shows $x;r = 0$
⟨*proof*⟩

lemma *start-points-path*:
assumes *backward-finite-path-root r x*
and *start-points x* $\neq$ *0*
shows *backward-terminating-path-root r x*
⟨*proof*⟩

lemma *root-in-start-points-2*:
assumes *backward-finite-path-root r x*
and *start-points x* $\neq$ *0*
shows $r \leq$ *start-points x*
⟨*proof*⟩

lemma *root-equals-start-points-2*:
assumes *backward-finite-path-root r x*
and *start-points x* $\neq$ *0*
shows $r =$ *start-points x*
⟨*proof*⟩

lemma *start-points-injective*:
assumes *backward-finite-path-root r x*
shows *is-inj* (*start-points x*)
⟨*proof*⟩

lemma *backward-terminating-path-root-aux-2*:
assumes *backward-finite-path-root r x*
and *start-points x* $\neq$ *0* $\vee$ $x = 0$
shows $x \leq x^{T\star};-(x^T;1)$
⟨*proof*⟩

lemma *start-points-not-zero-iff*:
assumes *backward-finite-path-root r x*
shows $x;r = 0 \wedge x \neq 0 \longleftrightarrow$ *start-points x* $\neq$ *0*
⟨*proof*⟩

Backwards terminating and backwards finite: Part II

lemma *backward-finite-path-root-acyclic-terminating-aux*:
assumes *backward-finite-path-root r x*
and *is-acyclic x*
shows $x;r = 0$
⟨*proof*⟩

lemma *backward-finite-path-root-acyclic-terminating-iff*:
assumes *backward-finite-path-root r x*

shows *is-acyclic* $x \longleftrightarrow x;r = 0$
⟨*proof*⟩

lemma *backward-finite-path-root-acyclic-terminating*:
assumes *backward-finite-path-root* r x
and *is-acyclic* x
shows *backward-terminating-path-root* r x
⟨*proof*⟩

lemma *non-empty-cycle-root-one-strongly-connected*:
assumes *non-empty-cycle-root* r x
shows *one-strongly-connected* x
⟨*proof*⟩

lemma *backward-finite-path-root-nodes-reachable*:
assumes *backward-finite-path-root* r x
and $v \leq x;1 + x^T;1$
and *is-sur* v
shows $r \leq x^\star;v$
⟨*proof*⟩

lemma *terminating-path-root-end-backward-terminating*:
assumes *terminating-path-root-end* r x e
shows *backward-terminating-path-root* r x
⟨*proof*⟩

lemma *terminating-path-root-end-converse*:
assumes *terminating-path-root-end* r x e
shows *terminating-path-root-end* e (x^T) r
⟨*proof*⟩

lemma *terminating-path-root-end-forward-terminating*:
assumes *terminating-path-root-end* r x e
shows *backward-terminating-path-root* e (x^T)
⟨*proof*⟩

end

3.3 Consequences with the Tarski rule and the point axiom

context *relation-algebra-rtc-tarski-point*
begin

Rooted paths

lemma *path-root-iff*:
$(\exists\, r$. *path-root* r $x) \longleftrightarrow$ *path* x
⟨*proof*⟩

lemma *non-empty-path-root-iff*:
$(\exists\, r$. *path-root* r $x \wedge r \leq (x + x^T);1) \longleftrightarrow$ *path* $x \wedge x \neq 0$

⟨*proof*⟩

(Non-empty) Cycle

lemma *non-empty-cycle-root-iff*:
$(\exists r\ .\ \textit{non-empty-cycle-root}\ r\ x) \longleftrightarrow \textit{cycle}\ x \wedge x \neq 0$
⟨*proof*⟩

lemma *non-empty-cycle-subset-equals*:
assumes *cycle x*
and *cycle y*
and $x \leq y$
and $x \neq 0$
shows $x = y$
⟨*proof*⟩

lemma *cycle-root-iff*:
$(\exists r\ .\ \textit{cycle-root}\ r\ x) \longleftrightarrow \textit{cycle}\ x$
⟨*proof*⟩

Backwards terminating and backwards finite

lemma *backward-terminating-path-root-iff*:
$(\exists r\ .\ \textit{backward-terminating-path-root}\ r\ x) \longleftrightarrow \textit{backward-terminating-path}\ x$
⟨*proof*⟩

lemma *non-empty-backward-terminating-path-root-iff*:
$\textit{backward-terminating-path-root}\ (\textit{start-points}\ x)\ x \longleftrightarrow \textit{backward-terminating-path}\ x \wedge x \neq 0$
⟨*proof*⟩

lemma *non-empty-backward-terminating-path-root-iff'*:
$\textit{backward-finite-path-root}\ (\textit{start-points}\ x)\ x \longleftrightarrow \textit{backward-terminating-path}\ x \wedge x \neq 0$
⟨*proof*⟩

lemma *backward-finite-path-root-iff*:
$(\exists r\ .\ \textit{backward-finite-path-root}\ r\ x) \longleftrightarrow \textit{backward-finite-path}\ x$
⟨*proof*⟩

lemma *non-empty-backward-finite-path-root-iff*:
$(\exists r\ .\ \textit{backward-finite-path-root}\ r\ x \wedge r \leq x;1) \longleftrightarrow \textit{backward-finite-path}\ x \wedge x \neq 0$
⟨*proof*⟩

Terminating

lemma *terminating-path-root-end-aux*:
assumes *terminating-path x*
shows $\exists r\ e\ .\ \textit{terminating-path-root-end}\ r\ x\ e$
⟨*proof*⟩

lemma *terminating-path-root-end-iff*:
$(\exists r\ e\ .\ \textit{terminating-path-root-end}\ r\ x\ e) \longleftrightarrow \textit{terminating-path}\ x$
⟨*proof*⟩

lemma *non-empty-terminating-path-root-end-iff*:
$\textit{terminating-path-root-end}\ (\textit{start-points}\ x)\ x\ (\textit{end-points}\ x) \longleftrightarrow \textit{terminating-path}\ x \wedge x \neq 0$
⟨*proof*⟩

lemma *non-empty-finite-path-root-end-iff*:
$\textit{finite-path-root-end}\ (\textit{start-points}\ x)\ x\ (\textit{end-points}\ x) \longleftrightarrow \textit{terminating-path}\ x \wedge x \neq 0$
⟨*proof*⟩

end

end

4 Correctness of Path Algorithms

To show that our theory of paths integrates with verification tasks, we verify the correctness of three basic path algorithms. Algorithms at the presented level are executable and can serve prototyping purposes. Data refinement can be carried out to move from such algorithms to more efficient programs. The total-correctness proofs use a library developed in [7].

theory *Path-Algorithms*

imports *HOL−Hoare.Hoare-Logic Rooted-Paths*

begin

unbundle *no trancl-syntax*

class *choose-singleton-point-signature* =
fixes *choose-singleton* :: $'a \Rightarrow 'a$
fixes *choose-point* :: $'a \Rightarrow 'a$

class *relation-algebra-rtc-tarski-choose-point* =
relation-algebra-rtc-tarski + *choose-singleton-point-signature* +
assumes *choose-singleton-singleton*: $x \neq 0 \Longrightarrow \textit{singleton}\ (\textit{choose-singleton}\ x)$
assumes *choose-singleton-decreasing*: $\textit{choose-singleton}\ x \leq x$
assumes *choose-point-point*: $\textit{is-vector}\ x \Longrightarrow x \neq 0 \Longrightarrow \textit{point}\ (\textit{choose-point}\ x)$
assumes *choose-point-decreasing*: $\textit{choose-point}\ x \leq x$
begin

no-notation
composition (**infixl** ‹;› *75*) **and**
times (**infixl** ‹∗› *70*)

notation
composition (**infixl** ‹∗› *75*)

4.1 Construction of a path

Our first example is a basic greedy algorithm that constructs a path from a vertex x to a different vertex y of a directed acyclic graph D.

abbreviation *construct-path-inv q x y D W* ≡
is-acyclic D ∧ *point x* ∧ *point y* ∧ *point q* ∧
$D^\star * q \le D^{T\star} * x$ ∧ $W \le D$ ∧ *terminating-path W* ∧
$(W = 0 \longleftrightarrow q{=}y) \wedge (W \ne 0 \longleftrightarrow q = \textit{start-points}\ W \wedge y = \textit{end-points}\ W)$

abbreviation *construct-path-inv-simp q x y D W* ≡
is-acyclic D ∧ *point x* ∧ *point y* ∧ *point q* ∧
$D^\star * q \le D^{T\star} * x$ ∧ $W \le D$ ∧ *terminating-path W* ∧
$q = \textit{start-points}\ W \wedge y = \textit{end-points}\ W$

lemma *construct-path-pre*:
assumes *is-acyclic D*
and *point y*
and *point x*
and $D^\star * y \le D^{T\star} * x$
shows *construct-path-inv y x y D 0*
⟨*proof*⟩

The following three lemmas are auxiliary lemmas for *construct-path-inv*. They are pulled out of the main proof to have more structure.

lemma *path-inv-points*:
assumes *construct-path-inv q x y D W* ∧ $q \ne x$
shows *point q*
and *point* (*choose-point* $(D{*}q)$)
⟨*proof*⟩

lemma *path-inv-choose-point-decrease*:
assumes *construct-path-inv q x y D W* ∧ $q \ne x$
shows $W{\ne}0 \Longrightarrow \textit{choose-point}\ (D{*}q) \le -((W + \textit{choose-point}\ (D{*}q) * q^T)^T{*}1)$
⟨*proof*⟩

lemma *end-points*:
assumes *construct-path-inv q x y D W* ∧ $q \ne x$
shows $\textit{choose-point}\ (D{*}q) = \textit{start-points}\ (W + \textit{choose-point}\ (D{*}q) * q^T)$
and $y = \textit{end-points}\ (W + \textit{choose-point}\ (D{*}q) * q^T)$
⟨*proof*⟩

lemma *construct-path-inv*:
assumes *construct-path-inv q x y D W* ∧ $q \ne x$

shows *construct-path-inv* (*choose-point* $(D*q)$) x y D (W + *choose-point* $(D*q)*q^T$)
⟨*proof*⟩

theorem *construct-path-partial*: *VARS* p q W
{ *is-acyclic* D ∧ *point* y ∧ *point* x ∧ $D^\star * y \leq D^{T\star} * x$ }
$W := 0$;
$q := y$;
WHILE $q \neq x$
INV { *construct-path-inv* q x y D W }
DO $p :=$ *choose-point* $(D*q)$;
$W := W + p*q^T$;
$q := p$
OD
{ $W \leq D$ ∧ *terminating-path* W ∧ ($W=0 \longleftrightarrow x=y$) ∧ ($W \neq 0 \longleftrightarrow x =$ *start-points* W ∧ $y =$ *end-points* W) }
⟨*proof*⟩

end

For termination, we additionally need finiteness.

context *finite*
begin

lemma *decrease-set*:
assumes $\forall x::'a$. Q $x \longrightarrow P$ x
and P w
and $\neg$ Q w
shows *card* { x . Q x } < *card* { x . P x }
⟨*proof*⟩

end

class *relation-algebra-rtc-tarski-choose-point-finite* =
relation-algebra-rtc-tarski-choose-point +
relation-algebra-rtc-tarski-point-finite
begin

lemma *decrease-variant*:
assumes $y \leq z$
and $w \leq z$
and $\neg$ $w \leq y$
shows *card* { x . $x \leq y$ } < *card* { x . $x \leq z$ }
⟨*proof*⟩

lemma *construct-path-inv-termination*:
assumes *construct-path-inv* q x y D W ∧ $q \neq x$
shows *card* { z . $z \leq -(W +$ *choose-point* $(D*q)*q^T)$ } < *card* { z . $z \leq -W$
}

⟨*proof*⟩

theorem *construct-path-total*: *VARS p q W*
[*is-acyclic D* $\wedge$ *point y* $\wedge$ *point x* $\wedge$ $D^\star * y \leq D^{T\star} * x$]
W := *0*;
q := *y*;
WHILE q $\neq$ *x*
INV { *construct-path-inv q x y D W* }
VAR { *card* { *z* . *z* $\leq -W$ } }
DO p := *choose-point* (*D*∗*q*);
W := *W* + $p * q^T$;
q := *p*
OD
[*W* $\leq$ *D* $\wedge$ *terminating-path W* $\wedge$ (*W*=*0* $\longleftrightarrow$ *x*=*y*) $\wedge$ (*W*$\neq$*0* $\longleftrightarrow$ *x* = *start-points W* $\wedge$ *y* = *end-points W*)]
⟨*proof*⟩

end

4.2 Topological sorting

In our second example we look at topological sorting. Given a directed acyclic graph, the problem is to construct a linear order of its vertices that contains x before y for each edge (x, y) of the graph. If the input graph models dependencies between tasks, the output is a linear schedule of the tasks that respects all dependencies.

context *relation-algebra-rtc-tarski-choose-point*
begin

abbreviation *topological-sort-inv*
where *topological-sort-inv q v R W* $\equiv$
regressively-finite R $\wedge$ $R \cdot v * v^T \leq W^+$ $\wedge$ *terminating-path W* $\wedge$ *W*∗*1* = $v \cdot -q$ $\wedge$
(*W* = *0* $\vee$ *q* = *end-points W*) $\wedge$ *point q* $\wedge$ $R * v \leq v$ $\wedge$ $q \leq v$ $\wedge$ *is-vector v*

lemma *topological-sort-pre*:
assumes *regressively-finite R*
shows *topological-sort-inv* (*choose-point* (*minimum R 1*)) (*choose-point* (*minimum R 1*)) *R 0*
⟨*proof*⟩

lemma *topological-sort-inv*:
assumes $v \neq 1$
and *topological-sort-inv q v R W*
shows *topological-sort-inv* (*choose-point* (*minimum R* (− *v*))) (*v* + *choose-point* (*minimum R* (− *v*))) *R* (*W* + *q* ∗ *choose-point* $(\textit{minimum } R\ (-\ v))^T$)

⟨*proof*⟩

lemma *topological-sort-post*:
assumes $\neg\ v \neq 1$
and *topological-sort-inv q v R W*
shows $R \leq W^{+} \wedge$ *terminating-path* $W \wedge (W + W^{T})*1 = -1'*1$
⟨*proof*⟩

theorem *topological-sort-partial*: *VARS p q v W*
{ *regressively-finite R* }
$W := 0$;
$q :=$ *choose-point* (*minimum R 1*);
$v := q$;
WHILE $v \neq 1$
INV { *topological-sort-inv q v R W* }
DO $p :=$ *choose-point* (*minimum R* $(-v)$);
$W := W + q*p^{T}$;
$q := p$;
$v := v + p$
OD
{ $R \leq W^{+} \wedge$ *terminating-path* $W \wedge (W + W^{T})*1 = -1'*1$ }
⟨*proof*⟩

end

context *relation-algebra-rtc-tarski-choose-point-finite*
begin

lemma *topological-sort-inv-termination*:
assumes $v \neq 1$
and *topological-sort-inv q v R W*
shows *card* $\{z\ .\ z \leq -(v +$ *choose-point* (*minimum R* $(-v)))\} <$ *card* $\{\ z\ .\ z \leq -v\ \}$
⟨*proof*⟩

Use precondition *is-acyclic* instead of *regressively-finite*. They are equivalent for finite graphs.

theorem *topological-sort-total*: *VARS p q v W*
[*is-acyclic R*]
$W := 0$;
$q :=$ *choose-point* (*minimum R 1*);
$v := q$;
WHILE $v \neq 1$
INV { *topological-sort-inv q v R W* }
VAR { *card* $\{\ z\ .\ z \leq -v\ \}$ }
DO $p :=$ *choose-point* (*minimum R* $(-v)$);
$W := W + q*p^{T}$;
$q := p$;
$v := v + p$

OD
$[\ R \leq W^+ \wedge terminating\text{-}path\ W \wedge (W + W^T)*1 = -1'*1\ ]$
⟨proof⟩

end

4.3 Construction of a tree

Our last application is a correctness proof of an algorithm that constructs a non-empty cycle for a given directed graph. This works in two steps. The first step is to construct a directed tree from a given root along the edges of the graph.

context *relation-algebra-rtc-tarski-choose-point*
begin

abbreviation *construct-tree-pre*
where $construct\text{-}tree\text{-}pre\ x\ y\ R \equiv y \leq R^{T\star}*x \wedge point\ x$
abbreviation *construct-tree-inv*
where $construct\text{-}tree\text{-}inv\ v\ x\ y\ D\ R \equiv construct\text{-}tree\text{-}pre\ x\ y\ R \wedge is\text{-}acyclic\ D \wedge is\text{-}inj\ D \wedge D \leq R \wedge D*x = 0 \wedge v = x + D^T*1 \wedge x*v^T \leq D^\star \wedge D \leq v*v^T \wedge is\text{-}vector\ v$
abbreviation *construct-tree-post*
where $construct\text{-}tree\text{-}post\ x\ y\ D\ R \equiv is\text{-}acyclic\ D \wedge is\text{-}inj\ D \wedge D \leq R \wedge D*x = 0 \wedge D^T*1 \leq D^{T\star}*x \wedge D^\star*y \leq D^{T\star}*x$

lemma *construct-tree-pre*:
assumes $construct\text{-}tree\text{-}pre\ x\ y\ R$
shows $construct\text{-}tree\text{-}inv\ x\ x\ y\ 0\ R$
⟨proof⟩

lemma *construct-tree-inv-aux*:
assumes $\neg\ y \leq v$
and $construct\text{-}tree\text{-}inv\ v\ x\ y\ D\ R$
shows $singleton\ (choose\text{-}singleton\ (v*-v^T \cdot R))$
⟨proof⟩

lemma *construct-tree-inv*:
assumes $\neg\ y \leq v$
and $construct\text{-}tree\text{-}inv\ v\ x\ y\ D\ R$
shows $construct\text{-}tree\text{-}inv\ (v + choose\text{-}singleton\ (v*-v^T \cdot R)^T*1)\ x\ y\ (D + choose\text{-}singleton\ (v*-v^T \cdot R))\ R$
⟨proof⟩

lemma *construct-tree-post*:
assumes $y \leq v$
and $construct\text{-}tree\text{-}inv\ v\ x\ y\ D\ R$

shows *construct-tree-post x y D R*
⟨*proof*⟩

theorem *construct-tree-partial*: *VARS e v D*
{ *construct-tree-pre x y R* }
D := *0*;
v := *x*;
WHILE $\neg$ *y* $\leq$ *v*
INV { *construct-tree-inv v x y D R* }
DO e := *choose-singleton* ($v{*}{-}v^T \cdot R$);
D := *D* + *e*;
v := *v* + $e^T{*}1$
OD
{ *construct-tree-post x y D R* }
⟨*proof*⟩

end

context *relation-algebra-rtc-tarski-choose-point-finite*
begin

lemma *construct-tree-inv-termination*:
assumes $\neg$ *y* $\leq$ *v*
and *construct-tree-inv v x y D R*
shows *card* { *z* . $z \leq -(v +$ *choose-singleton* $(v{*}{-}v^T \cdot R)^T{*}1)$ } < *card* { *z* . $z \leq -v$ }
⟨*proof*⟩

theorem *construct-tree-total*: *VARS e v D*
[*construct-tree-pre x y R*]
D := *0*;
v := *x*;
WHILE $\neg$ *y* $\leq$ *v*
INV { *construct-tree-inv v x y D R* }
VAR { *card* { *z* . $z \leq -v$ } }
DO e := *choose-singleton* ($v{*}{-}v^T \cdot R$);
D := *D* + *e*;
v := *v* + $e^T{*}1$
OD
[*construct-tree-post x y D R*]
⟨*proof*⟩

end

4.4 Construction of a non-empty cycle

The second step is to construct a path from the root to a given vertex in the tree. Adding an edge back to the root gives the cycle.

context *relation-algebra-rtc-tarski-choose-point*

begin

abbreviation *comment*
  where *comment* - ≡ *SKIP*
abbreviation *construct-cycle-inv*
  where *construct-cycle-inv v x y D R* ≡ *construct-tree-inv v x y D R* ∧ *point y* ∧ $y{*}x^T \leq R$

lemma *construct-cycle-pre*:
 assumes ¬ *is-acyclic R*
   and $y = choose\text{-}point\ ((R^+ \cdot 1')*1)$
   and $x = choose\text{-}point\ (R^\star*y \cdot R^T*y)$
  shows *construct-cycle-inv x x y 0 R*
⟨*proof*⟩

lemma *construct-cycle-pre2*:
 assumes $y \leq v$
   and *construct-cycle-inv v x y D R*
  shows *construct-path-inv y x y D 0* $\wedge D \leq R \wedge D * x = 0 \wedge y * x^T \leq R$
⟨*proof*⟩

lemma *construct-cycle-post*:
  assumes ¬ $q \neq x$
   and (*construct-path-inv q x y D W* $\wedge D \leq R \wedge D * x = 0 \wedge y * x^T \leq R$)
  shows $W + y * x^T \neq 0 \wedge W + y * x^T \leq R \wedge$ *cycle* $(W + y * x^T)$
⟨*proof*⟩

theorem *construct-cycle-partial*: *VARS e p q v x y C D W*
  { ¬ *is-acyclic R* }
  $y := choose\text{-}point\ ((R^+ \cdot 1')*1)$;
  $x := choose\text{-}point\ (R^\star*y \cdot R^T*y)$;
  $D := 0$;
  $v := x$;
  WHILE ¬ $y \leq v$
   INV { *construct-cycle-inv v x y D R* }
   DO $e := choose\text{-}singleton\ (v*-v^T \cdot R)$;
     $D := D + e$;
     $v := v + e^T*1$
   OD;
  comment { *is-acyclic D* ∧ *point y* ∧ *point x* ∧ $D^\star*y \leq D^{T\star}*x$ };
  $W := 0$;
  $q := y$;
  WHILE $q \neq x$
   INV { *construct-path-inv q x y D W* $\wedge D \leq R \wedge D*x = 0 \wedge y*x^T \leq R$ }
   DO $p := choose\text{-}point\ (D*q)$;
     $W := W + p*q^T$;
     $q := p$
   OD;
  comment { $W \leq D$ ∧ *terminating-path W* ∧ $(W = 0 \longleftrightarrow q=y) \wedge (W \neq 0$

$\longleftrightarrow$ *q* = *start-points W* ∧ *y* = *end-points W*) };
  C := *W* + $y{*}x^T$
  { *C* ≠ *0* ∧ *C* ≤ *R* ∧ *cycle C* }
  ⟨*proof*⟩

end

context *relation-algebra-rtc-tarski-choose-point-finite*
begin

theorem *construct-cycle-total*: *VARS e p q v x y C D W*
  [¬ *is-acyclic R*]
  y := *choose-point* $((R^+ \cdot 1')*1)$;
  x := *choose-point* $(R^\star * y \cdot R^T * y)$;
  D := *0*;
  v := *x*;
  WHILE ¬ *y* ≤ *v*
    INV { *construct-cycle-inv v x y D R* }
    VAR { *card* { *z* . *z* ≤ −*v* } }
     DO e := *choose-singleton* $(v{*}{-}v^T \cdot R)$;
        D := *D* + *e*;
        v := *v* + $e^T{*}1$
     OD;
  comment { *is-acyclic D* ∧ *point y* ∧ *point x* ∧ $D^\star{*}y \le D^{T\star}{*}x$ };
  W := *0*;
  q := *y*;
  WHILE q ≠ *x*
    INV { *construct-path-inv q x y D W* ∧ *D* ≤ *R* ∧ *D*∗*x* = *0* ∧ $y{*}x^T \le R$ }
    VAR { *card* { *z* . *z* ≤ − *W* } }
     DO p := *choose-point* (*D*∗*q*);
        W := *W* + $p{*}q^T$;
        q := *p*
     OD;
  comment { *W* ≤ *D* ∧ *terminating-path W* ∧ (*W* = *0* $\longleftrightarrow$ *q*=*y*) ∧ (*W* ≠ *0*
$\longleftrightarrow$ *q* = *start-points W* ∧ *y* = *end-points W*)};
  C := *W* + $y{*}x^T$
 [*C* ≠ *0* ∧ *C* ≤ *R* ∧ *cycle C*]
  ⟨*proof*⟩

end

end

References

[1] A. Armstrong, S. Foster, G. Struth, and T. Weber. Relation algebra. *Archive of Formal Proofs*, 2014.

[2] R. Berghammer, H. Furusawa, W. Guttmann, and P. Höfner. Relational characterisations of paths. *Journal of Logical and Algebraic Methods in Programming*, 117:100590, 2020.

[3] R. Diestel. *Graph Theory*. Springer, third edition, 2005.

[4] W. Guttmann. Stone-Kleene relation algebras. *Archive of Formal Proofs*, 2017.

[5] W. Guttmann. Stone relation algebras. *Archive of Formal Proofs*, 2017.

[6] W. Guttmann. An algebraic framework for minimum spanning tree problems. *Theoretical Comput. Sci.*, 744:37–55, 2018.

[7] W. Guttmann. Verifying minimum spanning tree algorithms with Stone relation algebras. *Journal of Logical and Algebraic Methods in Programming*, 101:132–150, 2018.

[8] D. Kozen. A completeness theorem for Kleene algebras and the algebra of regular events. *Information and Computation*, 110(2):366–390, 1994.

[9] K. C. Ng. *Relation Algebras with Transitive Closure*. PhD thesis, University of California, Berkeley, 1984.

[10] G. Schmidt and T. Ströhlein. *Relations and Graphs*. Springer, 1993.

[11] A. Tarski. On the calculus of relations. *The Journal of Symbolic Logic*, 6(3):73–89, 1941.

[12] G. Tinhofer. *Methoden der angewandten Graphentheorie*. Springer, 1976.