

Relational Forests

Walter Guttmann

June 17, 2024

Abstract

We study second-order formalisations of graph properties expressed as first-order formulas in relation algebras extended with a Kleene star. The formulas quantify over relations while still avoiding quantification over elements of the base set. We formalise the property of undirected graphs being acyclic this way. This involves a study of various kinds of orientation of graphs. We also verify basic algorithms to constructively prove several second-order properties.

Contents

1	Overview	1
2	Orientations and Undirected Forests	2
2.1	Orientability	2
2.2	Undirected forests	14
2.3	Arc axiom	32
2.4	Counterexamples	34
3	Axioms and Algorithmic Proofs	35
3.1	Constructing a spanning tree	36
3.2	Breadth-first search	41
3.3	Extending partial orders to linear orders	47

1 Overview

The theories described in this document study second-order specifications of graph properties expressed as first-order formulas in Stone-Kleene relation algebras. Of particular interest are undirected forests and their orientations, developed in Section 2. We also verify the correctness of a number of basic graph algorithms, which we use in constructive proofs of graph properties in Section 3.

The theories formally verify results in [5]; results from this paper are annotated with the corresponding theorem numbers. See the paper for further details and related work.

2 Orientations and Undirected Forests

In this theory we study orientations and various second-order specifications of undirected forests. The results are structured by the classes in which they can be proved, which correspond to algebraic structures. Most classes are generalisations of Kleene relation algebras. None of the classes except *kleene-relation-algebra* assumes the double-complement law $--x = x$ available in Boolean algebras. The corresponding paper does not elaborate these fine distinctions, so some results take a different form in this theory. They usually specialise to Kleene relation algebras after simplification using $--x = x$.

theory *Forests*

imports *Stone-Kleene-Relation-Algebras.Kleene-Relation-Algebras*

begin

2.1 Orientability

context *bounded-distrib-allegory-signature*

begin

abbreviation *irreflexive-inf* :: 'a \Rightarrow bool **where** *irreflexive-inf* x \equiv x \sqcap 1 = bot

end

context *bounded-distrib-allegory*

begin

lemma *irreflexive-inf-arc-asymmetric*:

irreflexive-inf x \implies arc x \implies asymmetric x

proof –

assume *irreflexive-inf* x arc x

hence bot = (x * top)^T \sqcap x

by (*metis arc-top-arc comp-right-one schroeder-1*)

thus ?thesis

by (*metis comp-inf.semiring.mult-zero-right conv-inf-bot-iff inf.sup-relative-same-increasing top-right-mult-increasing*)

qed

lemma *asymmetric-inf*:

asymmetric x \longleftrightarrow *irreflexive-inf* (x * x)

using *inf.sup-monoid.add-commute schroeder-2* **by** *force*

lemma *asymmetric-irreflexive-inf:*

asymmetric x \implies irreflexive-inf x

by (*metis asymmetric-inf-closed coreflexive-symmetric inf.idem inf-le2*)

lemma *transitive-asymmetric-irreflexive-inf:*

transitive x \implies asymmetric x \longleftrightarrow irreflexive-inf x

by (*smt asymmetric-inf asymmetric-irreflexive-inf inf.absorb2 inf.cobounded1 inf.sup-monoid.add-commute inf-assoc le-bot*)

abbreviation *orientation x y \equiv y \sqcup y^T = x \wedge asymmetric y*

abbreviation *loop-orientation x y \equiv y \sqcup y^T = x \wedge antisymmetric y*

abbreviation *super-orientation x y \equiv x \leq y \sqcup y^T \wedge asymmetric y*

abbreviation *loop-super-orientation x y \equiv x \leq y \sqcup y^T \wedge antisymmetric y*

lemma *orientation-symmetric:*

orientation x y \implies symmetric x

using *conv-dist-sup sup-commute* **by** *auto*

lemma *orientation-irreflexive-inf:*

orientation x y \implies irreflexive-inf x

using *asymmetric-irreflexive-inf asymmetric-conv-closed inf-sup-distrib2* **by** *auto*

lemma *loop-orientation-symmetric:*

loop-orientation x y \implies symmetric x

using *conv-dist-sup sup-commute* **by** *auto*

lemma *loop-orientation-diagonal:*

loop-orientation x y \implies y \sqcap y^T = x \sqcap 1

by (*metis inf.sup-monoid.add-commute inf.sup-same-context inf-le2 inf-sup-distrib1 one-inf-conv sup.idem*)

lemma *super-orientation-irreflexive-inf:*

super-orientation x y \implies irreflexive-inf x

using *coreflexive-bot-closed inf.sup-monoid.add-assoc inf.sup-right-divisibility inf-bot-right loop-orientation-diagonal* **by** *fastforce*

lemma *loop-super-orientation-diagonal:*

loop-super-orientation x y \implies x \sqcap 1 \leq y \sqcap y^T

using *inf.sup-right-divisibility inf-assoc loop-orientation-diagonal* **by** *fastforce*

definition *orientable x \equiv \exists y . orientation x y*

definition *loop-orientable x \equiv \exists y . loop-orientation x y*

definition *super-orientable x \equiv \exists y . super-orientation x y*

definition *loop-super-orientable x \equiv \exists y . loop-super-orientation x y*

lemma *orientable-symmetric:*

orientable $x \implies$ *symmetric* x
using *orientable-def orientation-symmetric* **by** *blast*

lemma *orientable-irreflexive-inf*:
orientable $x \implies$ *irreflexive-inf* x
using *orientable-def orientation-irreflexive-inf* **by** *blast*

lemma *loop-orientable-symmetric*:
loop-orientable $x \implies$ *symmetric* x
using *loop-orientable-def loop-orientation-symmetric* **by** *blast*

lemma *super-orientable-irreflexive-inf*:
super-orientable $x \implies$ *irreflexive-inf* x
using *super-orientable-def super-orientation-irreflexive-inf* **by** *blast*

lemma *orientable-down-closed*:
assumes *symmetric* x
and $x \leq y$
and *orientable* y
shows *orientable* x
proof –
from *assms(3)* **obtain** z **where** $1: z \sqcup z^T = y \wedge$ *asymmetric* z
using *orientable-def* **by** *blast*
let $?z = x \sqcap z$
have *orientation* x $?z$
proof (*rule conjI*)
show *asymmetric* $?z$
using 1 **by** (*simp add: conv-dist-inf inf.left-commute*
inf.sup-monoid.add-assoc)
thus $?z \sqcup ?z^T = x$
using 1 **by** (*metis assms(1,2) conv-dist-inf inf.orderE inf-sup-distrib1*)
qed
thus *thesis*
using *orientable-def* **by** *blast*
qed

lemma *loop-orientable-down-closed*:
assumes *symmetric* x
and $x \leq y$
and *loop-orientable* y
shows *loop-orientable* x
proof –
from *assms(3)* **obtain** z **where** $1: z \sqcup z^T = y \wedge$ *antisymmetric* z
using *loop-orientable-def* **by** *blast*
let $?z = x \sqcap z$
have *loop-orientation* x $?z$
proof (*rule conjI*)
show *antisymmetric* $?z$
using 1 *antisymmetric-inf-closed inf-commute* **by** *fastforce*

thus $?z \sqcup ?z^T = x$
using 1 **by** (*metis assms(1,2) conv-dist-inf inf.orderE inf-sup-distrib1*)
qed
thus *?thesis*
using *loop-orientable-def* **by** *blast*
qed

lemma *super-orientable-down-closed*:
assumes $x \leq y$
and *super-orientable y*
shows *super-orientable x*
using *assms order-lesseq-imp super-orientable-def* **by** *auto*

lemma *loop-super-orientable-down-closed*:
assumes $x \leq y$
and *loop-super-orientable y*
shows *loop-super-orientable x*
using *assms order-lesseq-imp loop-super-orientable-def* **by** *auto*

abbreviation *orientable-1* $x \equiv \text{loop-super-orientable } x$
abbreviation *orientable-2* $x \equiv \exists y . x \leq y \sqcup y^T \wedge y \sqcap y^T \leq x \sqcap 1$
abbreviation *orientable-3* $x \equiv \exists y . x \leq y \sqcup y^T \wedge y \sqcap y^T = x \sqcap 1$
abbreviation *orientable-4* $x \equiv \text{irreflexive-inf } x \longrightarrow \text{super-orientable } x$
abbreviation *orientable-5* $x \equiv \text{symmetric } x \longrightarrow \text{loop-orientable } x$
abbreviation *orientable-6* $x \equiv \text{symmetric } x \longrightarrow (\exists y . y \sqcup y^T = x \wedge y \sqcap y^T \leq x \sqcap 1)$
abbreviation *orientable-7* $x \equiv \text{symmetric } x \longrightarrow (\exists y . y \sqcup y^T = x \wedge y \sqcap y^T = x \sqcap 1)$
abbreviation *orientable-8* $x \equiv \text{symmetric } x \wedge \text{irreflexive-inf } x \longrightarrow \text{orientable } x$

lemma *super-orientation-diagonal*:
 $x \leq y \sqcup y^T \implies y \sqcap y^T \leq x \sqcap 1 \implies y \sqcap y^T = x \sqcap 1$
using *order.antisym loop-super-orientation-diagonal* **by** *auto*

lemma *orientable-2-implies-1*:
orientable-2 $x \implies \text{orientable-1 } x$
using *loop-super-orientable-def* **by** *auto*

lemma *orientable-2-3*:
orientable-2 $x \iff \text{orientable-3 } x$
using *eq-refl super-orientation-diagonal* **by** *blast*

lemma *orientable-5-6*:
orientable-5 $x \iff \text{orientable-6 } x$
using *loop-orientable-def loop-orientation-diagonal* **by** *fastforce*

lemma *orientable-6-7*:
orientable-6 $x \iff \text{orientable-7 } x$
using *super-orientation-diagonal* **by** *fastforce*

lemma *orientable-7-implies-8*:
orientable-7 $x \implies$ *orientable-8* x
using *orientable-def* **by** *blast*

lemma *orientable-5-implies-1*:
orientable-5 $(x \sqcup x^T) \implies$ *orientable-1* x
using *conv-dist-sup loop-orientable-def loop-super-orientable-def sup-commute*
by *fastforce*

ternary predicate S called *split* here

abbreviation *split* $x y z \equiv y \sqcap y^T = x \wedge y \sqcup y^T = z$

Theorem 3.1

lemma *orientation-split*:
orientation $x y \longleftrightarrow$ *split* *bot* $y x$
by *auto*

Theorem 3.2

lemma *split-1-loop-orientation*:
split $1 y x \implies$ *loop-orientation* $x y$
by *simp*

Theorem 3.3

lemma *loop-orientation-split*:
loop-orientation $x y \longleftrightarrow$ *split* $(x \sqcap 1) y x$
by (*metis inf.cobounded2 loop-orientation-diagonal*)

Theorem 3.4

lemma *loop-orientation-split-inf-1*:
loop-orientation $x y \longleftrightarrow$ *split* $(y \sqcap 1) y x$
by (*metis inf.sup-monoid.add-commute inf.sup-same-context inf-le2 one-inf-conv*)

lemma *loop-orientation-top-split*:
loop-orientation *top* $y \longleftrightarrow$ *split* $1 y$ *top*
by (*simp add: loop-orientation-split*)

injective and transitive orientations

definition *injectively-orientable* $x \equiv \exists y .$ *orientation* $x y \wedge$ *injective* y

lemma *injectively-orientable-orientable*:
injectively-orientable $x \implies$ *orientable* x
using *injectively-orientable-def orientable-def* **by** *auto*

lemma *orientable-orientable-1*:
orientable $x \implies$ *orientable-1* x
by (*metis bot-least order-refl loop-super-orientable-def orientable-def*)

lemma *injectively-orientable-down-closed*:
assumes *symmetric* x
and $x \leq y$
and *injectively-orientable* y
shows *injectively-orientable* x
proof –
from *assms*(3) **obtain** z **where** $1: z \sqcup z^T = y \wedge$ *asymmetric* $z \wedge$ *injective* z
using *injectively-orientable-def* **by** *blast*
let $?z = x \sqcap z$
have $2: \text{injective } ?z$
using 1 *inf-commute injective-inf-closed* **by** *fastforce*
have *orientation* x $?z$
proof (*rule conjI*)
show *asymmetric* $?z$
using 1 **by** (*simp add: conv-dist-inf inf.left-commute*
inf.sup-monoid.add-assoc)
thus $?z \sqcup ?z^T = x$
using 1 **by** (*metis assms(1,2) conv-dist-inf inf.orderE inf-sup-distrib1*)
qed
thus *?thesis*
using 2 *injectively-orientable-def* **by** *blast*
qed

definition *transitively-orientable* $x \equiv \exists y . \text{orientation } x y \wedge \text{transitive } y$

lemma *transitively-orientable-orientable*:
transitively-orientable $x \implies \text{orientable } x$
using *transitively-orientable-def orientable-def* **by** *auto*

lemma *irreflexive-transitive-orientation-asymmetric*:
assumes *irreflexive-inf* x
and *transitive* y
and $y \sqcup y^T = x$
shows *asymmetric* y
using *assms comp-inf.mult-right-dist-sup transitive-asymmetric-irreflexive-inf*
by *auto*

Theorem 12

lemma *transitively-orientable-2*:
transitively-orientable $x \iff \text{irreflexive-inf } x \wedge (\exists y . y \sqcup y^T = x \wedge \text{transitive } y)$
by (*metis irreflexive-transitive-orientation-asymmetric coreflexive-bot-closed*
loop-orientation-split transitively-orientable-def)

end

context *relation-algebra-signature*
begin

abbreviation *asymmetric-var* $:: 'a \Rightarrow \text{bool}$ **where** *asymmetric-var* $x \equiv$

irreflexive ($x * x$)

end

context *pd-allegory*

begin

[Theorem 1.4](#)

lemma *asymmetric-var*:

asymmetric $x \longleftrightarrow$ *asymmetric-var* x

using *asymmetric-inf pseudo-complement* **by** *auto*

[Theorem 1.3](#)

([Theorem 1.2](#) *is asymmetric-irreflexive* in *Relation-Algebras*)

lemma *transitive-asymmetric-irreflexive*:

transitive $x \implies$ *asymmetric* $x \longleftrightarrow$ *irreflexive* x

using *strict-order-var* **by** *blast*

lemma *orientable-irreflexive*:

orientable $x \implies$ *irreflexive* x

using *orientable-irreflexive-inf pseudo-complement* **by** *blast*

lemma *super-orientable-irreflexive*:

super-orientable $x \implies$ *irreflexive* x

using *pseudo-complement super-orientable-irreflexive-inf* **by** *blast*

lemma *orientation-diversity-split*:

orientation $(-1) y \longleftrightarrow$ *split bot* $y (-1)$

by *auto*

abbreviation *linear-orderable-1* $x \equiv$ *linear-order* x

abbreviation *linear-orderable-2* $x \equiv$ *linear-strict-order* x

abbreviation *linear-orderable-3* $x \equiv$ *transitive* $x \wedge$ *asymmetric* $x \wedge$ *strict-linear* x

abbreviation *linear-orderable-3a* $x \equiv$ *transitive* $x \wedge$ *strict-linear* x

abbreviation *orientable-11* $x \equiv$ *split 1* x *top*

abbreviation *orientable-12* $x \equiv$ *split bot* $x (-1)$

lemma *linear-strict-order-split*:

linear-strict-order $x \longleftrightarrow$ *transitive* $x \wedge$ *split bot* $x (-1)$

using *strict-order-var* **by** *blast*

[Theorem 1.6](#)

lemma *linear-strict-order-without-irreflexive*:

linear-strict-order $x \longleftrightarrow$ *transitive* $x \wedge$ *strict-linear* x

using *strict-linear-irreflexive* **by** *auto*

lemma *linear-order-without-reflexive*:

linear-order $x \longleftrightarrow$ *antisymmetric* $x \wedge$ *transitive* $x \wedge$ *linear* x


```

using linear-reflexive by blast

lemma linear-orderable-1-implies-2:
  linear-orderable-1  $x \implies$  linear-orderable-2 ( $x \sqcap -1$ )
  using linear-order-strict-order by blast

lemma linear-orderable-2-3:
  linear-orderable-2  $x \longleftrightarrow$  linear-orderable-3  $x$ 
  using linear-strict-order-split by auto

lemma linear-orderable-3-3a:
  linear-orderable-3  $x \longleftrightarrow$  linear-orderable-3a  $x$ 
  using strict-linear-irreflexive strict-order-var by blast

lemma linear-orderable-3-implies-orientable-12:
  linear-orderable-3  $x \implies$  orientable-12  $x$ 
  by simp

lemma orientable-11-implies-12:
  orientable-11  $x \implies$  orientable-12 ( $x \sqcap -1$ )
  by (smt inf-sup-distrib2 conv-complement conv-dist-inf conv-involutive
inf-import-p inf-top.left-neutral linear-asymmetric maddux-3-13 p-inf
symmetric-one-closed)

end

context stone-relation-algebra
begin

  Theorem 3.5

lemma split-symmetric-asymmetric:
  assumes regular  $x$ 
  shows split  $x \ y \ z \longleftrightarrow y \sqcap y^T = x \wedge (y \sqcap -y^T) \sqcup (y \sqcap -y^T)^T = z \sqcap -x \wedge x$ 
   $\leq z$ 
  proof
    assume split  $x \ y \ z$ 
    thus  $y \sqcap y^T = x \wedge (y \sqcap -y^T) \sqcup (y \sqcap -y^T)^T = z \sqcap -x \wedge x \leq z$ 
    by (metis conv-complement conv-dist-inf conv-involutive inf.cobounded1
inf.sup-monoid.add-commute inf-import-p inf-sup-distrib2 le-supI1)
  next
    assume  $y \sqcap y^T = x \wedge (y \sqcap -y^T) \sqcup (y \sqcap -y^T)^T = z \sqcap -x \wedge x \leq z$ 
    thus split  $x \ y \ z$ 
    by (smt (z3) assms conv-dist-sup conv-involutive inf.absorb2 inf.boundedE
inf.cobounded1 inf.idem inf.sup-monoid.add-commute inf-import-p
maddux-3-11-pp sup.left-commute sup-commute sup-inf-absorb)
  qed

lemma orientable-1-2:
  orientable-1  $x \longleftrightarrow$  orientable-2  $x$ 

```

proof
assume *orientable-1 x*
from this obtain y where $1: x \leq y \sqcup y^T \wedge y \sqcap y^T \leq 1$
using *loop-super-orientable-def by blast*
let $?y = (x \sqcap 1) \sqcup (y \sqcap -1)$
have $x \leq ?y \sqcup ?y^T \wedge ?y \sqcap ?y^T \leq x \sqcap 1$
proof
have $x \sqcap -1 \leq (y \sqcap -1) \sqcup (y^T \sqcap -1)$
using *1 inf.sup-right-divisibility inf-commute inf-left-commute inf-sup-distrib2 by auto*
also have $\dots \leq ?y \sqcup ?y^T$
by (*metis comp-inf.semiring.add-mono conv-complement conv-dist-inf conv-isotone sup.cobounded2 symmetric-one-closed*)
finally show $x \leq ?y \sqcup ?y^T$
by (*metis comp-inf.semiring.add-mono maddux-3-11-pp regular-one-closed sup.cobounded1 sup.left-idem*)
have $x = (x \sqcap 1) \sqcup (x \sqcap -1)$
by (*metis maddux-3-11-pp regular-one-closed*)
have $?y \sqcap ?y^T = (x \sqcap 1) \sqcup ((y \sqcap -1) \sqcap (y^T \sqcap -1))$
by (*metis comp-inf.semiring.distrib-left conv-complement conv-dist-inf conv-dist-sup coreflexive-symmetric distrib-imp1 inf-le2 symmetric-one-closed*)
also have $\dots = x \sqcap 1$
by (*metis 1 inf-assoc inf-commute pseudo-complement regular-one-closed selection-closed-id inf.cobounded2 maddux-3-11-pp*)
finally show $?y \sqcap ?y^T \leq x \sqcap 1$
by *simp*
qed
thus *orientable-2 x*
by *blast*
next
assume *orientable-2 x*
thus *orientable-1 x*
using *loop-super-orientable-def by auto*
qed

lemma *orientable-8-implies-5:*
assumes *orientable-8 (x \sqcap -1)*
shows *orientable-5 x*
proof
assume *1: symmetric x*
hence *symmetric (x \sqcap -1)*
by (*simp add: conv-complement symmetric-inf-closed*)
hence *orientable (x \sqcap -1)*
by (*simp add: assms pseudo-complement*)
from this obtain y where $2: y \sqcup y^T = x \sqcap -1 \wedge \text{asymmetric } y$
using *orientable-def by blast*
let $?y = y \sqcup (x \sqcap 1)$
have *loop-orientation x ?y*
proof

```

have ?y ⊔ ?yT = y ⊔ yT ⊔ (x ⊓ 1)
  using 1 conv-dist-inf conv-dist-sup sup-assoc sup-commute by auto
thus ?y ⊔ ?yT = x
  by (metis 2 maddux-3-11-pp regular-one-closed)
have ?y ⊓ ?yT = (y ⊓ yT) ⊔ (x ⊓ 1)
  by (simp add: 1 conv-dist-sup sup-inf-distrib2 symmetric-inf-closed)
thus antisymmetric ?y
  by (simp add: 2)
qed
thus loop-orientable x
  using loop-orientable-def by blast
qed

lemma orientable-4-implies-1:
  assumes orientable-4 (x ⊓ -1)
  shows orientable-1 x
proof -
  obtain y where 1: x ⊓ -1 ≤ y ⊔ yT ∧ asymmetric y
    using assms pseudo-complement super-orientable-def by auto
  let ?y = y ⊔ 1
  have loop-super-orientation x ?y
  proof
    show x ≤ ?y ⊔ ?yT
    by (smt 1 comp-inf.semiring.add-mono conv-dist-sup inf-le2 maddux-3-11-pp
    reflexive-one-closed regular-one-closed sup.absorb1 sup.left-commute sup-assoc
    symmetric-one-closed)
    show antisymmetric ?y
    using 1 conv-dist-sup distrib-imp1 inf-sup-distrib1 sup-monoid.add-commute
by auto
  qed
  thus ?thesis
  using loop-super-orientable-def by blast
qed

lemma orientable-1-implies-4:
  assumes orientable-1 (x ⊔ 1)
  shows orientable-4 x
proof
  assume 1: irreflexive-inf x
  obtain y where 2: x ⊔ 1 ≤ y ⊔ yT ∧ antisymmetric y
    using assms loop-super-orientable-def by blast
  let ?y = y ⊓ -1
  have super-orientation x ?y
  proof
    have x ≤ (y ⊔ yT) ⊓ -1
    using 1 2 pseudo-complement by auto
    thus x ≤ ?y ⊔ ?yT
    by (simp add: conv-complement conv-dist-inf inf-sup-distrib2)
    have ?y ⊓ ?yT = y ⊓ yT ⊓ -1

```

```

    using conv-complement conv-dist-inf inf-commute inf-left-commute by auto
  thus asymmetric ?y
    using 2 pseudo-complement by auto
qed
thus super-orientable x
  using super-orientable-def by blast
qed

```

lemma orientable-1-implies-5:

```

  assumes orientable-1 x
  shows orientable-5 x
proof
  assume 1: symmetric x
  obtain y where 2:  $x \leq y \sqcup y^T \wedge$  antisymmetric y
    using assms loop-super-orientable-def by blast
  let ?y =  $(x \sqcap 1) \sqcup (y \sqcap x \sqcap -1)$ 
  have loop-orientation x ?y
  proof
    have  $?y \sqcup ?y^T = ((y \sqcup y^T) \sqcap x \sqcap -1) \sqcup (x \sqcap 1)$ 
      by (simp add: 1 conv-complement conv-dist-inf conv-dist-sup inf-sup-distrib2
        sup.left-commute sup-commute)
    thus  $?y \sqcup ?y^T = x$ 
      by (metis 2 inf-absorb2 maddux-3-11-pp regular-one-closed)
    have  $?y \sqcap ?y^T = (x \sqcap 1) \sqcup ((y \sqcap x \sqcap -1) \sqcap (y^T \sqcap x^T \sqcap -1))$ 
      by (simp add: 1 conv-complement conv-dist-inf conv-dist-sup sup-inf-distrib1)
    thus antisymmetric ?y
      by (metis 2 antisymmetric-inf-closed conv-complement conv-dist-inf inf-le2
        le-supI symmetric-one-closed)
  qed
  thus loop-orientable x
    using loop-orientable-def by blast
qed

```

Theorem 2

lemma all-orientable-characterisations:

```

  shows  $(\forall x . \text{orientable-1 } x) \longleftrightarrow (\forall x . \text{orientable-2 } x)$ 
    and  $(\forall x . \text{orientable-1 } x) \longleftrightarrow (\forall x . \text{orientable-3 } x)$ 
    and  $(\forall x . \text{orientable-1 } x) \longleftrightarrow (\forall x . \text{orientable-4 } x)$ 
    and  $(\forall x . \text{orientable-1 } x) \longleftrightarrow (\forall x . \text{orientable-5 } x)$ 
    and  $(\forall x . \text{orientable-1 } x) \longleftrightarrow (\forall x . \text{orientable-6 } x)$ 
    and  $(\forall x . \text{orientable-1 } x) \longleftrightarrow (\forall x . \text{orientable-7 } x)$ 
    and  $(\forall x . \text{orientable-1 } x) \longleftrightarrow (\forall x . \text{orientable-8 } x)$ 
  subgoal using orientable-1-2 by simp
  subgoal using orientable-1-2 orientable-2-3 by simp
  subgoal using orientable-1-implies-4 orientable-4-implies-1 by blast
  subgoal using orientable-5-implies-1 orientable-1-implies-5 by blast
  subgoal using orientable-5-6 orientable-5-implies-1 orientable-1-implies-5 by
blast
  subgoal using orientable-5-6 orientable-5-implies-1 orientable-6-7

```

orientable-1-implies-5 **by force**
subgoal using *orientable-5-6 orientable-5-implies-1 orientable-6-7*
orientable-7-implies-8 orientable-1-implies-5 orientable-8-implies-5 **by auto**
done

lemma *orientable-12-implies-11*:
orientable-12 $x \implies$ *orientable-11* $(x \sqcup 1)$
by (*smt inf-top.right-neutral conv-complement conv-dist-sup conv-involutive*
inf-import-p maddux-3-13 p-bot p-dist-inf p-dist-sup regular-one-closed
symmetric-one-closed)

lemma *linear-strict-order-order*:
linear-strict-order $x \implies$ *linear-order* $(x \sqcup 1)$
by (*simp add: strict-order-order transitive-asymmetric-irreflexive*
orientable-12-implies-11)

lemma *linear-orderable-2-implies-1*:
linear-orderable-2 $x \implies$ *linear-orderable-1* $(x \sqcup 1)$
using *linear-strict-order-order* **by simp**

[Theorem 4](#)

[Theorem 12](#)

[Theorem 13](#)

lemma *exists-split-characterisations*:
shows $(\exists x . \text{linear-orderable-1 } x) \longleftrightarrow (\exists x . \text{linear-orderable-2 } x)$
and $(\exists x . \text{linear-orderable-1 } x) \longleftrightarrow (\exists x . \text{linear-orderable-3 } x)$
and $(\exists x . \text{linear-orderable-1 } x) \longleftrightarrow (\exists x . \text{linear-orderable-3a } x)$
and $(\exists x . \text{linear-orderable-1 } x) \longleftrightarrow \text{transitively-orientable } (-1)$
and $(\exists x . \text{linear-orderable-1 } x) \implies (\exists x . \text{orientable-11 } x)$
and $(\exists x . \text{orientable-11 } x) \longleftrightarrow (\exists x . \text{orientable-12 } x)$
subgoal 1 using *linear-strict-order-order linear-order-strict-order* **by blast**
subgoal 2 using *1 strict-order-var* **by blast**
subgoal using *1 linear-strict-order-without-irreflexive* **by auto**
subgoal using *2 transitively-orientable-def* **by auto**
subgoal using *loop-orientation-top-split* **by blast**
subgoal using *orientable-11-implies-12 orientable-12-implies-11* **by blast**
done

[Theorem 4](#)

[Theorem 12](#)

lemma *exists-all-orientable*:
shows $(\exists x . \text{orientable-11 } x) \longleftrightarrow (\forall x . \text{orientable-1 } x)$
and $\text{transitively-orientable } (-1) \implies (\forall x . \text{orientable-8 } x)$
subgoal apply (*rule iffI*)
subgoal using *loop-super-orientable-def top-greatest* **by blast**
subgoal using *loop-orientation-top-split loop-super-orientable-def top-le* **by**
blast

```

    done
  subgoal using orientable-down-closed pseudo-complement
    transitively-orientable-orientable by blast
  done
end

```

2.2 Undirected forests

We start with a few general results in Kleene algebras and a few basic properties of directed acyclic graphs.

```

context kleene-algebra
begin

```

Theorem 1.9

```

lemma plus-separate-comp-bot:
  assumes  $x * y = \text{bot}$ 
  shows  $(x \sqcup y)^+ = x^+ \sqcup y^+ \sqcup y^+ * x^+$ 
proof -
  have  $(x \sqcup y)^+ = x * y^* * x^* \sqcup y^+ * x^*$ 
  using assms cancel-separate-1 semiring.distrib-right mult-assoc by auto
  also have  $\dots = x^+ \sqcup y^+ * x^*$ 
  by (simp add: assms star-absorb)
  finally show ?thesis
  by (metis star.circ-back-loop-fixpoint star.circ-plus-same sup-assoc
    sup-commute mult-assoc)
qed
end

```

```

context bounded-distrib-kleene-allegory
begin

```

```

lemma reflexive-inf-plus-star:
  assumes reflexive  $x$ 
  shows  $x \sqcap y^+ \leq 1 \iff x \sqcap y^* = 1$ 
  using assms reflexive-inf-star sup.absorb-iff1 by auto
end

```

```

context pd-kleene-allegory
begin

```

```

lemma acyclic-star-inf-conv-iff:
  assumes irreflexive  $w$ 
  shows  $\text{acyclic } w \iff w^* \sqcap w^{T^*} = 1$ 

```

by (*metis assms acyclic-star-below-complement-1 acyclic-star-inf-conv conv-complement conv-order equivalence-one-closed inf.absorb1 inf.left-commute pseudo-complement star.circ-increasing*)

Theorem 1.7

lemma *acyclic-irreflexive-star-antisymmetric:*

acyclic $x \longleftrightarrow$ *irreflexive* $x \wedge$ *antisymmetric* (x^*)

by (*metis acyclic-star-inf-conv-iff conv-star-commute order.trans*

reflexive-inf-closed star.circ-mult-increasing star.circ-reflexive order.antisym)

Theorem 1.8

lemma *acyclic-plus-asymmetric:*

acyclic $x \longleftrightarrow$ *asymmetric* (x^+)

using *acyclic-asymmetric asymmetric-irreflexive star.circ-transitive-equal*

star.left-plus-circ mult-assoc **by** *auto*

Theorem 1.3

(Theorem 1.1 is *acyclic-asymmetric* in *Kleene-Relation-Algebras*)

lemma *transitive-acyclic-irreflexive:*

transitive $x \implies$ *acyclic* $x \longleftrightarrow$ *irreflexive* x

using *order.antisym star.circ-mult-increasing star-right-induct-mult* **by** *fastforce*

lemma *transitive-acyclic-asymmetric:*

transitive $x \implies$ *acyclic* $x \longleftrightarrow$ *asymmetric* x

using *strict-order-var transitive-acyclic-irreflexive* **by** *blast*

Theorem 1.5

lemma *strict-order-transitive-acyclic:*

strict-order $x \longleftrightarrow$ *transitive* $x \wedge$ *acyclic* x

using *transitive-acyclic-irreflexive* **by** *auto*

lemma *linear-strict-order-transitive-acyclic:*

linear-strict-order $x \longleftrightarrow$ *transitive* $x \wedge$ *acyclic* $x \wedge$ *strict-linear* x

using *transitive-acyclic-irreflexive* **by** *auto*

The following are various specifications of an undirected graph being acyclic.

definition *acyclic-1* $x \equiv \forall y . \text{orientation } x y \longrightarrow \text{acyclic } y$

definition *acyclic-1b* $x \equiv \forall y . \text{orientation } x y \longrightarrow y^* \sqcap y^{T*} = 1$

definition *acyclic-2* $x \equiv \forall y . y \leq x \wedge \text{asymmetric } y \longrightarrow \text{acyclic } y$

definition *acyclic-2a* $x \equiv \forall y . y \sqcup y^T \leq x \wedge \text{asymmetric } y \longrightarrow \text{acyclic } y$

definition *acyclic-2b* $x \equiv \forall y . y \sqcup y^T \leq x \wedge \text{asymmetric } y \longrightarrow y^* \sqcap y^{T*} = 1$

definition *acyclic-3a* $x \equiv \forall y . y \leq x \wedge x \leq y^* \longrightarrow y = x$

definition *acyclic-3b* $x \equiv \forall y . y \leq x \wedge y^* = x^* \longrightarrow y = x$

definition *acyclic-3c* $x \equiv \forall y . y \leq x \wedge x \leq y^+ \longrightarrow y = x$

definition *acyclic-3d* $x \equiv \forall y . y \leq x \wedge y^+ = x^+ \longrightarrow y = x$

definition *acyclic-4* $x \equiv \forall y . y \leq x \longrightarrow x \sqcap y^* \leq \text{--}y$

definition *acyclic-4a* $x \equiv \forall y . y \leq x \longrightarrow x \sqcap y^* \leq y$

definition *acyclic-4b* $x \equiv \forall y . y \leq x \longrightarrow x \sqcap y^* = y$
definition *acyclic-4c* $x \equiv \forall y . y \leq x \longrightarrow y \sqcap (x \sqcap -y)^* = \text{bot}$
definition *acyclic-5a* $x \equiv \forall y . y \leq x \longrightarrow y^* \sqcap (x \sqcap -y)^* = 1$
definition *acyclic-5b* $x \equiv \forall y . y \leq x \longrightarrow y^* \sqcap (x \sqcap -y)^+ \leq 1$
definition *acyclic-5c* $x \equiv \forall y . y \leq x \longrightarrow y^+ \sqcap (x \sqcap -y)^* \leq 1$
definition *acyclic-5d* $x \equiv \forall y . y \leq x \longrightarrow y^+ \sqcap (x \sqcap -y)^+ \leq 1$
definition *acyclic-5e* $x \equiv \forall y z . y \leq x \wedge z \leq x \wedge y \sqcap z = \text{bot} \longrightarrow y^* \sqcap z^* = 1$
definition *acyclic-5f* $x \equiv \forall y z . y \sqcup z \leq x \wedge y \sqcap z = \text{bot} \longrightarrow y^* \sqcap z^* = 1$
definition *acyclic-5g* $x \equiv \forall y z . y \sqcup z = x \wedge y \sqcap z = \text{bot} \longrightarrow y^* \sqcap z^* = 1$
definition *acyclic-6* $x \equiv \exists y . y \sqcup y^T = x \wedge \text{acyclic } y \wedge \text{injective } y$

Theorem 6

lemma *acyclic-2-2a*:

assumes *symmetric* x

shows *acyclic-2* $x \longleftrightarrow \text{acyclic-2a } x$

proof –

have $\bigwedge y . y \leq x \longleftrightarrow y \sqcup y^T \leq x$

using *assms conv-isotone* **by force**

thus *?thesis*

by (*simp add: acyclic-2-def acyclic-2a-def*)

qed

Theorem 6

lemma *acyclic-2a-2b*:

shows *acyclic-2a* $x \longleftrightarrow \text{acyclic-2b } x$

by (*simp add: acyclic-2a-def acyclic-2b-def acyclic-star-inf-conv-iff asymmetric-irreflexive*)

Theorem 5

lemma *acyclic-1-1b*:

shows *acyclic-1* $x \longleftrightarrow \text{acyclic-1b } x$

by (*simp add: acyclic-1-def acyclic-1b-def acyclic-star-inf-conv-iff asymmetric-irreflexive*)

Theorem 10

lemma *acyclic-6-1-injectively-orientable*:

acyclic-6 $x \longleftrightarrow \text{acyclic-1 } x \wedge \text{injectively-orientable } x$

proof

assume *acyclic-6* x

from this obtain y **where** $1: y \sqcup y^T = x \wedge \text{acyclic } y \wedge \text{injective } y$

using *acyclic-6-def* **by blast**

have *acyclic-1* x

proof (*unfold acyclic-1-def, rule allI, rule impI*)

fix z

assume $2: \text{orientation } x z$

hence $3: z = (z \sqcap y) \sqcup (z \sqcap y^T)$

by (*metis 1 inf-sup-absorb inf-sup-distrib1*)

have $(z \sqcap y) * (z \sqcap y^T) \leq z * z \sqcap y * y^T$

by (*simp add: comp-isotone*)

also have $\dots \leq -1 \sqcap 1$
using *1 2 asymmetric-var comp-inf.mult-isotone* **by** *blast*
finally have $\downarrow: (z \sqcap y) * (z \sqcap y^T) = \text{bot}$
by (*simp add: le-bot*)
have $z^+ = (z \sqcap y)^+ \sqcup (z \sqcap y^T)^+ \sqcup (z \sqcap y^T)^+ * (z \sqcap y)^+$
using *3 4 plus-separate-comp-bot* **by** *fastforce*
also have $\dots \leq y^+ \sqcup (z \sqcap y^T)^+ \sqcup (z \sqcap y^T)^+ * (z \sqcap y)^+$
using *comp-isotone semiring.add-right-mono star-isotone* **by** *auto*
also have $\dots \leq y^+ \sqcup y^{T+} \sqcup (z \sqcap y^T)^+ * (z \sqcap y)^+$
using *comp-isotone semiring.add-left-mono semiring.add-right-mono*
star-isotone **by** *auto*
also have $\dots \leq -1 \sqcup (z \sqcap y^T)^+ * (z \sqcap y)^+$
by (*smt 1 conv-complement conv-isotone conv-plus-commute inf.absorb2*
inf.orderE order-conv-closed order-one-closed semiring.add-right-mono
sup.absorb1)
also have $\dots = -1$
proof –
have $(z \sqcap y^T)^+ * (z \sqcap y)^+ \leq (z \sqcap y^T) * \text{top} * (z \sqcap y)^+$
using *comp-isotone* **by** *auto*
also have $\dots \leq (z \sqcap y^T) * \text{top} * (z \sqcap y)$
by (*metis inf.eq-refl star.circ-left-top star-plus mult-assoc*)
also have $\dots \leq -1$
by (*metis 4 bot-least comp-commute-below-diversity inf.absorb2*
pseudo-complement schroeder-1 mult-assoc)
finally show *?thesis*
using *sup.absorb1* **by** *blast*
qed
finally show *acyclic z*
by *simp*
qed
thus *acyclic-1 x \wedge injectively-orientable x*
using *1 injectively-orientable-def acyclic-asymmetric* **by** *blast*
next
assume *acyclic-1 x \wedge injectively-orientable x*
thus *acyclic-6 x*
using *acyclic-6-def acyclic-1-def injectively-orientable-def* **by** *auto*
qed

lemma *acyclic-6-symmetric:*
acyclic-6 x \implies symmetric x
by (*simp add: acyclic-6-1-injectively-orientable injectively-orientable-orientable*
orientable-symmetric)

lemma *acyclic-6-irreflexive:*
acyclic-6 x \implies irreflexive x
by (*simp add: acyclic-6-1-injectively-orientable injectively-orientable-orientable*
orientable-irreflexive)

lemma *acyclic-4-irreflexive:*

acyclic-4 $x \implies$ *irreflexive* x
by (*metis acyclic-4-def bot-least inf.absorb2 inf.sup-monoid.add-assoc p-bot pseudo-complement star.circ-reflexive*)

Theorem 6.4

lemma *acyclic-2-implies-1*:
acyclic-2 $x \implies$ *acyclic-1* x
using *acyclic-2-def acyclic-1-def* **by** *auto*

Theorem 8

lemma *acyclic-4a-4b*:
acyclic-4a $x \iff$ *acyclic-4b* x
using *acyclic-4a-def acyclic-4b-def order.eq-iff star.circ-increasing* **by** *auto*

Theorem 7

lemma *acyclic-3a-3b*:
acyclic-3a $x \iff$ *acyclic-3b* x
by (*metis acyclic-3a-def acyclic-3b-def order.antisym star.circ-increasing star-involutive star-isotone*)

Theorem 7

lemma *acyclic-3a-3c*:
assumes *irreflexive* x
shows *acyclic-3a* $x \iff$ *acyclic-3c* x
proof
assume *acyclic-3a* x
thus *acyclic-3c* x
by (*meson acyclic-3a-def acyclic-3c-def order-lesseq-imp star.left-plus-below-circ*)
next
assume 1: *acyclic-3c* x
show *acyclic-3a* x
proof (*unfold acyclic-3a-def, rule allI, rule impI*)
fix y
assume $y \leq x \wedge x \leq y^*$
hence $y \leq x \wedge x \leq y^+$
by (*metis assms inf.order-lesseq-imp le-infI p-inf-sup-below star-left-unfold-equal*)
thus $y = x$
using 1 *acyclic-3c-def* **by** *blast*
qed
qed

Theorem 7

lemma *acyclic-3c-3d*:
shows *acyclic-3c* $x \iff$ *acyclic-3d* x
proof –
have $\bigwedge y z . y \leq z \wedge z \leq y^+ \iff y \leq z \wedge y^+ = z^+$
apply (*rule iffI*)

apply (*smt comp-associative plus-sup star.circ-transitive-equal*
star.left-plus-circ sup-absorb1 sup-absorb2)
by (*simp add: star.circ-mult-increasing*)
thus *?thesis*
by (*simp add: acyclic-3c-def acyclic-3d-def*)
qed

Theorem 8

lemma *acyclic-4a-implies-3a*:
acyclic-4a x \implies acyclic-3a x
using *acyclic-4a-def acyclic-3a-def inf.absorb1* **by** *auto*

lemma *acyclic-4a-implies-4*:
acyclic-4a x \implies acyclic-4 x
by (*simp add: acyclic-4-def acyclic-4a-4b acyclic-4b-def pp-increasing*)

lemma *acyclic-4b-implies-4c*:
acyclic-4b x \implies acyclic-4c x
by (*simp add: acyclic-4b-def acyclic-4c-def inf.sup-relative-same-increasing*)

Theorem 8.5

lemma *acyclic-4-implies-2*:
assumes *symmetric x*
shows *acyclic-4 x \implies acyclic-2 x*
proof –
assume *1: acyclic-4 x*
show *acyclic-2 x*
proof (*unfold acyclic-2-def, rule allI, rule impI*)
fix *y*
assume *2: y \leq x \wedge asymmetric y*
hence *y^T \leq x \sqcap \neg y*
using *assms conv-inf-bot-iff conv-isotone pseudo-complement* **by** *force*
hence *y^{*} \sqcap y^T \leq y^{*} \sqcap x \sqcap \neg y*
using *dual-order.trans* **by** *auto*
also have *... \leq \neg \neg y \sqcap \neg y*
using *1 2* **by** (*metis inf commute acyclic-4-def comp-inf.mult-left-isotone*)
finally show *acyclic y*
by (*simp add: acyclic-star-below-complement-1 le-bot*)
qed
qed

Theorem 10.3

lemma *acyclic-6-implies-4a*:
acyclic-6 x \implies acyclic-4a x
proof –
assume *acyclic-6 x*
from this obtain *y* **where** *1: y \sqcup y^T = x \wedge acyclic y \wedge injective y*
using *acyclic-6-def* **by** *auto*
show *acyclic-4a x*

proof (*unfold acyclic-4a-def, rule allI, rule impI*)
fix z
assume $z \leq x$
hence $z = (z \sqcap y) \sqcup (z \sqcap y^T)$
using 1 by (*metis inf.orderE inf-sup-distrib1*)
hence 2: $z^* = (z \sqcap y^T)^* * (z \sqcap y)^*$
using 1 by (*metis cancel-separate-2*)
hence $x \sqcap z^* = (y \sqcap (z \sqcap y^T)^* * (z \sqcap y)^*) \sqcup (y^T \sqcap (z \sqcap y^T)^* * (z \sqcap y)^*)$
using 1 inf-sup-distrib2 by auto
also have $\dots \leq z$
proof (*rule sup-least*)
have $y \sqcap (z \sqcap y^T)^* * (z \sqcap y)^* = (y \sqcap (z \sqcap y^T)^*) \sqcup (y \sqcap z^* * (z \sqcap y))$
using 2 by (*metis inf-sup-distrib1 star.circ-back-loop-fixpoint sup-commute*)
also have $\dots \leq (y \sqcap y^{T*}) \sqcup (y \sqcap z^* * (z \sqcap y))$
using *inf.sup-right-isotone semiring.add-right-mono star-isotone* **by auto**
also have $\dots = y \sqcap z^* * (z \sqcap y)$
using 1 by (*metis acyclic-star-below-complement bot-least*
inf.sup-monoid.add-commute pseudo-complement sup.absorb2)
also have $\dots \leq (z^* \sqcap y * (z \sqcap y^T)^*) * (z \sqcap y)$
using *dedekind-2 inf-commute* **by auto**
also have $\dots \leq y * y^T * z$
by (*simp add: conv-isotone inf.coboundedI2 mult-isotone*)
also have $\dots \leq z$
using 1 mult-left-isotone **by fastforce**
finally show $y \sqcap (z \sqcap y^T)^* * (z \sqcap y)^* \leq z$
·
have $y^T \sqcap (z \sqcap y^T)^* * (z \sqcap y)^* = (y^T \sqcap (z \sqcap y)^*) \sqcup (y^T \sqcap (z \sqcap y^T)^* * z^*)$
using 2 by (*metis inf-sup-distrib1 star.circ-loop-fixpoint sup-commute*)
also have $\dots \leq (y^T \sqcap y^*) \sqcup (y^T \sqcap (z \sqcap y^T)^* * z^*)$
using *inf.sup-right-isotone semiring.add-right-mono star-isotone* **by auto**
also have $\dots = y^T \sqcap (z \sqcap y^T)^* * z^*$
using 1 acyclic-star-below-complement-1 inf-commute **by auto**
also have $\dots \leq (z \sqcap y^T)^* * (z^* \sqcap (z \sqcap y^T)^T * y^T)$
using *dedekind-1 inf-commute* **by auto**
also have $\dots \leq z * y * y^T$
by (*simp add: comp-associative comp-isotone conv-dist-inf inf.coboundedI2*)
also have $\dots \leq z$
using 1 mult-right-isotone mult-assoc **by fastforce**
finally show $y^T \sqcap (z \sqcap y^T)^* * (z \sqcap y)^* \leq z$
·
qed
finally show $x \sqcap z^* \leq z$
·
qed
qed

Theorem 1.10

lemma *top-injective-inf-complement:*

assumes *injective x*

shows $top * (x \sqcap y) \sqcap top * (x \sqcap -y) = bot$
proof –
have $(x \sqcap -y) * (x^T \sqcap y^T) \leq -1$
by (*metis conv-dist-inf inf.cobounded2 inf-left-idem mult-left-one p-shunting-swap schroeder-4-p*)
hence $(x \sqcap -y) * (x^T \sqcap y^T) = bot$
by (*smt assms comp-isotone coreflexive-comp-inf coreflexive-idempotent coreflexive-symmetric dual-order.trans inf.cobounded1 strict-order-var*)
thus *?thesis*
by (*simp add: conv-dist-inf schroeder-2 mult-assoc*)
qed

lemma *top-injective-inf-complement-2:*

assumes *injective x*
shows $(x^T \sqcap y) * top \sqcap (x^T \sqcap -y) * top = bot$
by (*smt assms top-injective-inf-complement conv-dist-comp conv-dist-inf conv-involutive conv-complement conv-top conv-bot*)

Theorem 10.3

lemma *acyclic-6-implies-5a:*

acyclic-6 x \implies acyclic-5a x

proof –

assume *acyclic-6 x*
from this obtain y where 1: $y \sqcup y^T = x \wedge acyclic\ y \wedge injective\ y$
using *acyclic-6-def by auto*
show *acyclic-5a x*
proof (*unfold acyclic-5a-def, rule allI, rule impI*)
fix z
assume $z \leq x$
hence 2: $z = (z \sqcap y) \sqcup (z \sqcap y^T)$
by (*metis 1 inf.orderE inf-sup-distrib1*)
hence 3: $z^* = (z \sqcap y^T)^* * (z \sqcap y)^*$
by (*metis 1 cancel-separate-2*)
have $(x \sqcap -z)^* = ((y \sqcap -z) \sqcup (y^T \sqcap -z))^*$
using 1 inf-sup-distrib2 by auto
also have $\dots = (y^T \sqcap -z)^* * (y \sqcap -z)^*$
using 1 cancel-separate-2 inf-commute by auto
finally have $z^* \sqcap (x \sqcap -z)^* = (y^T \sqcap z)^* * (y \sqcap z)^* \sqcap (y^T \sqcap -z)^* * (y \sqcap -z)^*$
using 3 inf-commute by simp
also have $\dots = ((y \sqcap z)^* \sqcap (y^T \sqcap -z)^* * (y \sqcap -z)^*) \sqcup ((y^T \sqcap z)^+ * (y \sqcap z)^* \sqcap (y^T \sqcap -z)^* * (y \sqcap -z)^*)$
by (*smt inf.sup-monoid.add-commute inf-sup-distrib1 star.circ-loop-fixpoint sup-commute mult-assoc*)
also have $\dots = (1 \sqcap (y^T \sqcap -z)^* * (y \sqcap -z)^*) \sqcup ((y \sqcap z)^+ \sqcap (y^T \sqcap -z)^* * (y \sqcap -z)^*) \sqcup ((y^T \sqcap z)^+ * (y \sqcap z)^* \sqcap (y^T \sqcap -z)^* * (y \sqcap -z)^*)$
by (*metis inf-sup-distrib2 star-left-unfold-equal*)
also have $\dots \leq 1$
proof (*intro sup-least*)
show $1 \sqcap (y^T \sqcap -z)^* * (y \sqcap -z)^* \leq 1$

by simp
have $(y \sqcap z)^+ \sqcap (y^T \sqcap -z)^* * (y \sqcap -z)^* = ((y \sqcap z)^+ \sqcap (y^T \sqcap -z)^*) \sqcup ((y \sqcap z)^+ \sqcap (y^T \sqcap -z)^* * (y \sqcap -z)^+)$
by (*metis inf-sup-distrib1 star.circ-back-loop-fixpoint star.circ-plus-same sup-commute mult-assoc*)
also have $\dots \leq \text{bot}$
proof (*rule sup-least*)
have $(y \sqcap z)^+ \sqcap (y^T \sqcap -z)^* \leq y^+ \sqcap y^{T*}$
by (*meson comp-inf.mult-isotone comp-isotone inf.cobounded1 star-isotone*)
also have $\dots = \text{bot}$
using 1 by (*smt acyclic-star-inf-conv inf.orderE inf.sup-monoid.add-assoc pseudo-complement star.left-plus-below-circ*)
finally show $(y \sqcap z)^+ \sqcap (y^T \sqcap -z)^* \leq \text{bot}$
 \cdot
have $(y \sqcap z)^+ \sqcap (y^T \sqcap -z)^* * (y \sqcap -z)^+ \leq \text{top} * (y \sqcap z) \sqcap \text{top} * (y \sqcap -z)$
by (*metis comp-associative comp-inf.mult-isotone star.circ-left-top star.circ-plus-same top-left-mult-increasing*)
also have $\dots = \text{bot}$
using 1 by (*simp add: top-injective-inf-complement*)
finally show $(y \sqcap z)^+ \sqcap (y^T \sqcap -z)^* * (y \sqcap -z)^+ \leq \text{bot}$
 \cdot
qed
finally show $(y \sqcap z)^+ \sqcap (y^T \sqcap -z)^* * (y \sqcap -z)^* \leq 1$
using bot-least le-bot by blast
have $(y^T \sqcap z)^+ * (y \sqcap z)^* \sqcap (y^T \sqcap -z)^* * (y \sqcap -z)^* = ((y^T \sqcap z)^+ * (y \sqcap z)^* \sqcap (y^T \sqcap -z)^* * (y \sqcap -z)^*) \sqcup ((y^T \sqcap z)^+ * (y \sqcap z)^* \sqcap (y^T \sqcap -z)^+ * (y \sqcap -z)^*)$
by (*metis inf-sup-distrib1 star.circ-loop-fixpoint sup-commute mult-assoc*)
also have $\dots = ((y^T \sqcap z)^+ * (y \sqcap z)^* \sqcap 1) \sqcup ((y^T \sqcap z)^+ * (y \sqcap z)^* \sqcap (y \sqcap -z)^+) \sqcup ((y^T \sqcap z)^+ * (y \sqcap z)^* \sqcap (y^T \sqcap -z)^+ * (y \sqcap -z)^*)$
by (*metis inf-sup-distrib1 star-left-unfold-equal*)
also have $\dots \leq 1$
proof (*intro sup-least*)
show $(y^T \sqcap z)^+ * (y \sqcap z)^* \sqcap 1 \leq 1$
by simp
have $(y^T \sqcap z)^+ * (y \sqcap z)^* \sqcap (y \sqcap -z)^+ = ((y^T \sqcap z)^+ \sqcap (y \sqcap -z)^+) \sqcup ((y^T \sqcap z)^+ * (y \sqcap z)^+ \sqcap (y \sqcap -z)^+)$
by (*smt inf.sup-monoid.add-commute inf-sup-distrib1 star.circ-back-loop-fixpoint star.circ-plus-same sup-commute mult-assoc*)
also have $\dots \leq \text{bot}$
proof (*rule sup-least*)
have $(y^T \sqcap z)^+ \sqcap (y \sqcap -z)^+ \leq y^{T+} \sqcap y^+$
by (*meson comp-inf.mult-isotone comp-isotone inf.cobounded1 star-isotone*)
also have $\dots = \text{bot}$
using 1 by (*metis acyclic-asymmetric conv-inf-bot-iff conv-plus-commute star-sup-1 sup.idem mult-assoc*)
finally show $(y^T \sqcap z)^+ \sqcap (y \sqcap -z)^+ \leq \text{bot}$
 \cdot

have $(y^T \sqcap z)^+ * (y \sqcap z)^+ \sqcap (y \sqcap -z)^+ \leq \text{top} * (y \sqcap z) \sqcap \text{top} * (y \sqcap -z)$
by (*smt comp-inf.mult-isotone comp-isotone inf.cobounded1 inf.orderE*
star.circ-plus-same top.extremum mult-assoc)
also have $\dots = \text{bot}$
using 1 **by** (*simp add: top-injective-inf-complement*)
finally show $(y^T \sqcap z)^+ * (y \sqcap z)^+ \sqcap (y \sqcap -z)^+ \leq \text{bot}$
qed
finally show $(y^T \sqcap z)^+ * (y \sqcap z)^* \sqcap (y \sqcap -z)^+ \leq 1$
using *bot-least le-bot by blast*
have $(y^T \sqcap z)^+ * (y \sqcap z)^* \sqcap (y^T \sqcap -z)^+ * (y \sqcap -z)^* \leq (y^T \sqcap z) * \text{top} \sqcap$
 $(y^T \sqcap -z) * \text{top}$
using *comp-associative inf.sup-mono mult-right-isotone top.extremum by*
presburger
also have $\dots = \text{bot}$
using 1 **by** (*simp add: top-injective-inf-complement-2*)
finally show $(y^T \sqcap z)^+ * (y \sqcap z)^* \sqcap (y^T \sqcap -z)^+ * (y \sqcap -z)^* \leq 1$
using *bot-least le-bot by blast*
qed
finally show $(y^T \sqcap z)^+ * (y \sqcap z)^* \sqcap (y^T \sqcap -z)^* * (y \sqcap -z)^* \leq 1$
qed
finally show $z^* \sqcap (x \sqcap -z)^* = 1$
by (*simp add: order.antisym star.circ-reflexive*)
qed
qed

Theorem 9.7

lemma *acyclic-5b-implies-4*:
assumes *irreflexive x*
and *acyclic-5b x*
shows *acyclic-4 x*
proof (*unfold acyclic-4-def, rule allI, rule impI*)
fix y
assume $y \leq x$
hence $y^* \sqcap (x \sqcap -y)^+ \leq 1$
using *acyclic-5b-def assms(2) by blast*
hence $y^* \sqcap x \sqcap -y \leq 1$
by (*smt inf.sup-left-divisibility inf.sup-monoid.add-assoc*
star.circ-mult-increasing)
hence $y^* \sqcap x \sqcap -y = \text{bot}$
by (*smt assms(1) comp-inf.semiring.mult-zero-left inf.orderE*
inf.sup-monoid.add-assoc inf.sup-monoid.add-commute pseudo-complement)
thus $x \sqcap y^* \leq --y$
using *inf.sup-monoid.add-commute pseudo-complement by fastforce*
qed

Theorem 9

lemma *acyclic-5a-5b*:

acyclic-5a $x \longleftrightarrow$ *acyclic-5b* x
by (*simp add: acyclic-5a-def acyclic-5b-def star.circ-reflexive reflexive-inf-plus-star*)

Theorem 9

lemma *acyclic-5a-5c*:
acyclic-5a $x \longleftrightarrow$ *acyclic-5c* x
by (*metis acyclic-5a-def acyclic-5c-def inf-commute star.circ-reflexive reflexive-inf-plus-star*)

Theorem 9

lemma *acyclic-5b-5d*:
acyclic-5b $x \longleftrightarrow$ *acyclic-5d* x
proof –
have *acyclic-5b* $x \longleftrightarrow (\forall y . y \leq x \longrightarrow (y^+ \sqcup 1) \sqcap (x \sqcap -y)^+ \leq 1)$
by (*simp add: acyclic-5b-def star-left-unfold-equal sup-commute*)
also have ... \longleftrightarrow *acyclic-5d* x
by (*simp add: inf-sup-distrib2 acyclic-5d-def*)
finally show *?thesis*

qed

lemma *acyclic-5a-5e*:
acyclic-5a $x \longleftrightarrow$ *acyclic-5e* x
proof
assume *1: acyclic-5a* x
show *acyclic-5e* x
proof (*unfold acyclic-5e-def, intro allI, rule impI*)
fix $y z$
assume *2: $y \leq x \wedge z \leq x \wedge y \sqcap z = \text{bot}$*
hence $z \leq x \sqcap -y$
using *p-antitone-iff pseudo-complement* **by** *auto*
hence $y^* \sqcap z^* \leq 1$
using *1 2* **by** (*metis acyclic-5a-def comp-inf.mult-isotone inf.cobounded1 inf.right-idem star-isotone*)
thus $y^* \sqcap z^* = 1$
by (*simp add: order.antisym star.circ-reflexive*)

qed

next
assume *1: acyclic-5e* x
show *acyclic-5a* x
proof (*unfold acyclic-5a-def, rule allI, rule impI*)
fix y
let $?z = x \sqcap -y$
assume *2: $y \leq x$*
have $y \sqcap ?z = \text{bot}$
by (*simp add: inf.left-commute*)
thus $y^* \sqcap ?z^* = 1$
using *1 2* **by** (*simp add: acyclic-5e-def*)

qed
qed

Theorem 9

lemma *acyclic-5e-5f*:
acyclic-5e $x \longleftrightarrow$ *acyclic-5f* x
by (*simp add: acyclic-5e-def acyclic-5f-def*)

lemma *acyclic-5e-down-closed*:
assumes $x \leq y$
and *acyclic-5e* y
shows *acyclic-5e* x
using *assms acyclic-5e-def order.trans* **by** *blast*

lemma *acyclic-5a-down-closed*:
assumes $x \leq y$
and *acyclic-5a* y
shows *acyclic-5a* x
using *acyclic-5e-down-closed assms acyclic-5a-5e* **by** *blast*

further variants of the existence of a linear order

abbreviation *linear-orderable-4* $x \equiv$ *transitive* $x \wedge$ *acyclic* $x \wedge$ *strict-linear* x
abbreviation *linear-orderable-5* $x \equiv$ *transitive* $x \wedge$ *acyclic* $x \wedge$ *linear* (x^*)
abbreviation *linear-orderable-6* $x \equiv$ *acyclic* $x \wedge$ *linear* (x^*)
abbreviation *linear-orderable-7* $x \equiv$ *split 1* (x^*) *top*
abbreviation *linear-orderable-8* $x \equiv$ *split bot* (x^*) (-1)

lemma *linear-orderable-3-4*:
linear-orderable-3 $x \longleftrightarrow$ *linear-orderable-4* x
using *transitive-acyclic-asymmetric* **by** *blast*

lemma *linear-orderable-5-implies-6*:
linear-orderable-5 $x \implies$ *linear-orderable-6* x
by *simp*

lemma *linear-orderable-6-implies-3*:
assumes *linear-orderable-6* x
shows *linear-orderable-3* (x^+)

proof –
have 1: *transitive* (x^+)
by (*simp add: comp-associative mult-isotone star.circ-mult-upper-bound star.left-plus-below-circ*)
have 2: *asymmetric* (x^+)
by (*simp add: assms acyclic-asymmetric star.circ-transitive-equal star.left-plus-circ mult-assoc*)
have 3: *strict-linear* (x^+)
by (*smt assms acyclic-star-inf-conv conv-star-commute inf.sup-monoid.add-commute inf-absorb2 maddux-3-13 orientable-11-implies-12 star-left-unfold-equal*)

show *?thesis*
using 1 2 3 **by** *simp*
qed

lemma *linear-orderable-7-implies-1:*
linear-orderable-7 $x \implies$ *linear-orderable-1* (x^*)
using *star.circ-transitive-equal* **by** *auto*

lemma *linear-orderable-6-implies-8:*
linear-orderable-6 $x \implies$ *linear-orderable-8* x
by (*simp add: linear-orderable-6-implies-3*)

abbreviation *path-orderable* $x \equiv$ *univalent* $x \wedge$ *injective* $x \wedge$ *acyclic* $x \wedge$ *linear* (x^*)

lemma *path-orderable-implies-linear-orderable-6:*
path-orderable $x \implies$ *linear-orderable-6* x
by *simp*

definition *simple-paths* $x \equiv \exists y . y \sqcup y^T = x \wedge$ *acyclic* $y \wedge$ *injective* $y \wedge$ *univalent* y

Theorem 14.1

lemma *simple-paths-acyclic-6:*
simple-paths $x \implies$ *acyclic-6* x
using *simple-paths-def acyclic-6-def* **by** *blast*

Theorem 14.2

lemma *simple-paths-transitively-orientable:*
assumes *simple-paths* x
shows *transitively-orientable* ($x^+ \sqcap -1$)

proof –

from *assms* **obtain** y **where** 1: $y \sqcup y^T = x \wedge$ *acyclic* $y \wedge$ *injective* $y \wedge$ *univalent* y

using *simple-paths-def* **by** *auto*

let $?y = y^+$

have 2: *transitive* $?y$

by (*simp add: comp-associative mult-right-isotone star.circ-mult-upper-bound star.left-plus-below-circ*)

have 3: *asymmetric* $?y$

using 1 *acyclic-plus-asymmetric* **by** *auto*

have $?y \sqcup ?y^T = x^+ \sqcap -1$

proof (*rule order.antisym*)

have 4: $?y \leq x^+$

using 1 *comp-isotone star-isotone* **by** *auto*

hence $?y^T \leq x^+$

using 1 **by** (*metis conv-dist-sup conv-involutive conv-order conv-plus-commute sup-commute*)

thus $?y \sqcup ?y^T \leq x^+ \sqcap -1$

using 1 4 **by** (*simp add: irreflexive-conv-closed*)
have $x^+ \leq y^* \sqcup y^{*T}$
using 1 **by** (*metis cancel-separate-1-sup conv-star-commute*
star.left-plus-below-circ)
also have $\dots = ?y \sqcup ?y^T \sqcup 1$
by (*smt conv-plus-commute conv-star-commute star.circ-reflexive*
star.left-unfold-equal sup.absorb1 sup-assoc sup-monoid.add-commute)
finally show $x^+ \sqcap -1 \leq ?y \sqcup ?y^T$
by (*metis inf.order-lesseq-imp inf.sup-monoid.add-commute*
inf.sup-right-isotone p-inf-sup-below sup-commute)
qed
thus *?thesis*
using 2 3 *transitively-orientable-def* **by** *auto*
qed

abbreviation *spanning* $x \ y \equiv y \leq x \wedge x \leq (y \sqcup y^T)^* \wedge \text{acyclic } y \wedge \text{injective } y$
definition *spannable* $x \equiv \exists y . \text{spanning } x \ y$

lemma *acyclic-6-implies-spannable*:
acyclic-6 $x \implies \text{spannable } x$
by (*metis acyclic-6-def star.circ-increasing sup.cobounded1 spannable-def*)

lemma *acyclic-3a-spannable-implies-6*:
assumes *acyclic-3a* x
and *spannable* x
and *symmetric* x
shows *acyclic-6* x
by (*smt acyclic-6-def acyclic-3a-def assms conv-isotone le-supI spannable-def*)

Theorem 10.3

lemma *acyclic-6-implies-3a*:
acyclic-6 $x \implies \text{acyclic-3a } x$
by (*simp add: acyclic-6-implies-4a acyclic-4a-implies-3a*)

Theorem 10.3

lemma *acyclic-6-implies-2*:
acyclic-6 $x \implies \text{acyclic-2 } x$
by (*simp add: acyclic-6-implies-4a acyclic-6-symmetric acyclic-4-implies-2*
acyclic-4a-implies-4)

Theorem 11

lemma *acyclic-6-3a-spannable*:
acyclic-6 $x \iff \text{symmetric } x \wedge \text{spannable } x \wedge \text{acyclic-3a } x$
using *acyclic-6-implies-3a acyclic-3a-spannable-implies-6*
acyclic-6-implies-spannable acyclic-6-symmetric **by** *blast*

end

context *stone-kleene-relation-algebra*

begin

Theorem 11.3

lemma *point-spanning*:

assumes *point* p

shows *spanning* (-1) $(p \sqcap -1)$

spannable (-1)

proof –

let $?y = p \sqcap -1$

have 1 : *injective* $?y$

by (*simp add: assms injective-inf-closed*)

have $?y * ?y \leq -1$

using *assms cancel-separate-5 inf.sup-monoid.add-commute vector-inf-comp*

by *auto*

hence 2 : *transitive* $?y$

by (*simp add: assms vector-inf-comp*)

hence 3 : *acyclic* $?y$

by (*simp add: transitive-acyclic-irreflexive*)

have 4 : $p \leq ?y \sqcup 1$

by (*simp add: regular-complement-top sup-commute sup-inf-distrib1*)

have $top = p^T * p$

using *assms order.eq-iff shunt-bijective top-greatest vector-conv-covector* **by**

blast

also have $\dots \leq (?y \sqcup 1)^T * (?y \sqcup 1)$

using 4 **by** (*simp add: conv-isotone mult-isotone*)

also have $\dots = (?y \sqcup ?y^T)^*$

using 1 2 **by** (*smt order.antisym cancel-separate-1 conv-star-commute*

star.circ-mult-1 star.circ-mult-increasing star.right-plus-circ

star-right-induct-mult sup-commute)

finally have $-1 \leq (?y \sqcup ?y^T)^*$

using *top.extremum top-le* **by** *blast*

thus *spanning* (-1) $(p \sqcap -1)$

using 1 3 *inf.cobounded2* **by** *blast*

thus *spannable* (-1)

using *spannable-def* **by** *blast*

qed

lemma *irreflexive-star*:

$(x \sqcap -1)^* = x^*$

proof –

have 1 : $x \sqcap 1 \leq (x \sqcap -1)^*$

by (*simp add: le-infI2 star.circ-reflexive*)

have $x \sqcap -1 \leq (x \sqcap -1)^*$

by (*simp add: star.circ-increasing*)

hence $x \leq (x \sqcap -1)^*$

using 1 **by** (*smt maddux-3-11-pp regular-one-closed sup.absorb-iff1 sup-assoc*)

thus *?thesis*

by (*metis order.antisym inf.cobounded1 star-involutive star-isotone*)

qed

Theorem 6.5

lemma *acyclic-2-1*:

assumes *orientable x*

shows *acyclic-2 x* \longleftrightarrow *acyclic-1 x*

proof

assume *acyclic-2 x*

thus *acyclic-1 x*

using *acyclic-2-implies-1* by *blast*

next

assume 1: *acyclic-1 x*

obtain *y* where 2: *orientation x y* \wedge *symmetric x*

using *assms orientable-def orientable-symmetric* by *blast*

show *acyclic-2 x*

proof (*unfold acyclic-2-def, rule allI, rule impI*)

fix *z*

assume 3: $z \leq x \wedge$ *asymmetric z*

let $?z = (-z \sqcap x) \sqcup (-z \sqcup z^T) \sqcap y$

have *orientation x ?z*

proof

have $?z \sqcup ?z^T = ((-z \sqcup -z^T) \sqcap x) \sqcup (-z \sqcup z^T) \sqcap (y \sqcup y^T)$

by (*smt 2 3 comp-inf.semiring.combine-common-factor conv-complement conv-dist-inf conv-dist-sup inf-sup-distrib1 orientation-symmetric sup.left-commute sup-assoc*)

also have $\dots = x$

by (*metis 2 inf-commute maddux-3-11-pp pp-dist-sup sup-monoid.add-commute*)

finally show $?z \sqcup ?z^T = x$

·

have $?z \sqcap ?z^T = ((-z \sqcap x) \sqcup (-z \sqcup z^T) \sqcap y) \sqcap ((-z^T \sqcap x) \sqcup (-z \sqcup z^T) \sqcap y^T)$

by (*simp add: 2 conv-complement conv-dist-inf conv-dist-sup inf.sup-monoid.add-commute*)

also have $\dots = ((-z \sqcap x) \sqcap (-z^T \sqcap x)) \sqcup ((-z \sqcap x) \sqcap (-z \sqcup z^T) \sqcap y^T) \sqcup ((-z \sqcup z^T) \sqcap y) \sqcap (-z^T \sqcap x) \sqcup ((-z \sqcup z^T) \sqcap y) \sqcap (-z \sqcup z^T) \sqcap y^T$

by (*smt comp-inf.semiring.distrib-left inf-sup-distrib2 sup-assoc*)

also have $\dots = \text{bot}$

by (*smt 2 3 inf.cobounded1 inf.left-commute inf.orderE p-dist-sup pseudo-complement sup.absorb-iff1*)

finally show $?z \sqcap ?z^T = \text{bot}$

·

qed

hence 4: *acyclic ?z*

using 1 *acyclic-1-def* by *auto*

have $z \leq ?z$

by (*simp add: 3 le-supI1 pp-increasing*)

thus *acyclic z*

using 4 *comp-isotone star-isotone* by *fastforce*

qed
qed

Theorem 8

lemma *acyclic-4-4c*:

$acyclic-4\ x \longleftrightarrow acyclic-4c\ x$

proof

assume 1: *acyclic-4* x

show *acyclic-4c* x

proof (*unfold acyclic-4c-def, rule allI, rule impI*)

fix y

assume 2: $y \leq x$

have $x \sqcap (x \sqcap -y)^* \leq --(x \sqcap -y)$

using 1 *acyclic-4-def inf.cobounded1* by *blast*

also have $\dots \leq -y$

by *simp*

finally have $x \sqcap y \sqcap (x \sqcap -y)^* = bot$

by (*simp add: p-shunting-swap pseudo-complement*)

thus $y \sqcap (x \sqcap -y)^* = bot$

using 2 *inf-absorb2* by *auto*

qed

next

assume 3: *acyclic-4c* x

show *acyclic-4* x

proof (*unfold acyclic-4-def, rule allI, rule impI*)

fix y

assume 4: $y \leq x$

have $x \sqcap -y \sqcap (x \sqcap -(x \sqcap -y))^* = bot$

using 3 *acyclic-4c-def inf-le1* by *blast*

hence $x \sqcap -y \sqcap (x \sqcap --y)^* = bot$

using *inf-import-p* by *auto*

hence $x \sqcap -y \sqcap (x \sqcap y)^* = bot$

by (*smt p-inf-pp pp-dist-star pp-pp-inf-bot-iff*)

hence $x \sqcap -y \sqcap y^* = bot$

using 4 *inf-absorb2* by *auto*

thus $x \sqcap y^* \leq --y$

using *p-shunting-swap pseudo-complement* by *auto*

qed

qed

Theorem 9

lemma *acyclic-5f-5g*:

$acyclic-5f\ x \longleftrightarrow acyclic-5g\ x$

proof

assume *acyclic-5f* x

thus *acyclic-5g* x

using *acyclic-5f-def acyclic-5g-def* by *auto*

next

assume 1: *acyclic-5g* x

```

show acyclic-5f x
proof (unfold acyclic-5f-def, intro allI, rule impI)
  fix y z
  let ?y = x  $\sqcap$  --y
  let ?z = x  $\sqcap$  -y
  assume y  $\sqcup$  z  $\leq$  x  $\wedge$  y  $\sqcap$  z = bot
  hence y  $\leq$  ?y  $\wedge$  z  $\leq$  ?z
    using inf.sup-monoid.add-commute pseudo-complement by fastforce
  hence y*  $\sqcap$  z*  $\leq$  ?y*  $\sqcap$  ?z*
    using comp-inf.mult-isotone star-isotone by blast
  also have ... = 1
    using 1 by (simp add: acyclic-5g-def inf.left-commute
inf.sup-monoid.add-commute maddux-3-11-pp)
  finally show y*  $\sqcap$  z* = 1
    by (simp add: order.antisym star.circ-reflexive)
qed
qed

```

```

lemma linear-orderable-3-implies-5:
  assumes linear-orderable-3 x
  shows linear-orderable-5 x
proof –
  have top = x  $\sqcup$  xT  $\sqcup$  1
    using assms conv-dist-sup orientable-12-implies-11 sup-assoc sup-commute by
fastforce
  also have ...  $\leq$  x*  $\sqcup$  x*T
    by (smt conv-star-commute star.circ-increasing star-sup-one sup-assoc
sup-commute sup-mono)
  finally show ?thesis
    by (simp add: assms top-le transitive-acyclic-asymmetric)
qed

```

```

lemma linear-orderable-8-implies-7:
  linear-orderable-8 x  $\implies$  linear-orderable-7 x
  using orientable-12-implies-11 star-left-unfold-equal sup-commute by fastforce

```

Theorem 13

```

lemma exists-split-characterisations-2:
  shows  $(\exists x . \text{linear-orderable-1 } x) \longleftrightarrow (\exists x . \text{linear-orderable-4 } x)$ 
  and  $(\exists x . \text{linear-orderable-1 } x) \longleftrightarrow (\exists x . \text{linear-orderable-5 } x)$ 
  and  $(\exists x . \text{linear-orderable-1 } x) \longleftrightarrow (\exists x . \text{linear-orderable-6 } x)$ 
  and  $(\exists x . \text{linear-orderable-1 } x) \longleftrightarrow (\exists x . \text{linear-orderable-7 } x)$ 
  and  $(\exists x . \text{linear-orderable-1 } x) \longleftrightarrow (\exists x . \text{linear-orderable-8 } x)$ 
  subgoal 1 using exists-split-characterisations(1) strict-order-transitive-acyclic
by auto
  subgoal 2 using 1 linear-orderable-3-implies-5 linear-orderable-6-implies-3
transitive-acyclic-asymmetric by auto
  subgoal 3 using 2 exists-split-characterisations(1) linear-orderable-6-implies-3
by auto

```

```

  subgoal using 2 linear-orderable-8-implies-7 linear-orderable-6-implies-3
  linear-orderable-7-implies-1 by blast
  subgoal using 3 linear-orderable-8-implies-7 asymmetric-irreflexive
  linear-orderable-6-implies-3 by blast
  done

end

```

2.3 Arc axiom

```

class stone-kleene-relation-algebra-arc = stone-kleene-relation-algebra +
  assumes arc-axiom:  $x \neq \text{bot} \implies \exists y . \text{arc } y \wedge y \leq \text{--}x$ 
begin

```

```

subclass stone-relation-algebra-tarski
proof unfold-locales
  fix x
  assume 1: regular x and 2:  $x \neq \text{bot}$ 
  from 2 obtain y where  $\text{arc } y \wedge y \leq \text{--}x$ 
  using arc-axiom by auto
  thus  $\text{top} * x * \text{top} = \text{top}$ 
  using 1 by (metis mult-assoc le-iff-sup mult-left-isotone semiring.distrib-left
  sup.orderE top.extremum)
qed

```

```

context
  assumes orientable-path:  $\text{arc } x \implies x \leq \text{--}y^* \implies \exists z . z \leq y \wedge \text{asymmetric } z$ 
   $\wedge x \leq \text{--}z^*$ 
begin

```

Theorem 8.6

```

lemma acyclic-2-4:
  assumes irreflexive x
  and symmetric x
  shows  $\text{acyclic-2 } x \iff \text{acyclic-4 } x$ 
proof
  show  $\text{acyclic-2 } x \implies \text{acyclic-4 } x$ 
  proof (unfold acyclic-4-def, intro allI, intro impI, rule ccontr)
    fix y
    assume 1:  $\text{acyclic-2 } x$  and 2:  $y \leq x$  and 3:  $\neg x \sqcap y^* \leq \text{--}y$ 
    hence  $x \sqcap y^* \sqcap \text{--}y \neq \text{bot}$ 
    by (simp add: pseudo-complement)
    from this obtain z where 4:  $\text{arc } z \wedge z \leq \text{--}(x \sqcap y^* \sqcap \text{--}y)$ 
    using arc-axiom by blast
    from this obtain w where 5:  $w \leq y \wedge \text{asymmetric } w \wedge z \leq \text{--}w^*$ 
    using orientable-path by auto
    let ?y =  $w \sqcup (z^T \sqcap x)$ 
    have 6:  $?y \leq x$ 
    using 2 5 by auto
  end
end

```



```

have ?y  $\sqcap$  ?yT = (w  $\sqcap$  wT)  $\sqcup$  (w  $\sqcap$  z  $\sqcap$  xT)  $\sqcup$  (zT  $\sqcap$  x  $\sqcap$  wT)  $\sqcup$  (zT  $\sqcap$  x  $\sqcap$  z
 $\sqcap$  xT)
  by (simp add: inf.commute sup.commute inf.left-commute sup.left-commute
conv-dist-inf conv-dist-sup inf-sup-distrib1)
also have ...  $\leq$  bot
proof (intro sup-least)
  show w  $\sqcap$  wT  $\leq$  bot
    by (simp add: 5)
  have w  $\sqcap$  z  $\sqcap$  xT  $\leq$  y  $\sqcap$  z
    by (simp add: 5 inf.coboundedI1)
  also have ...  $\leq$  y  $\sqcap$  -y
    using 4 by (metis eq-refl inf.cobounded1 inf.left-commute
inf.sup-monoid.add-commute inf-p order-trans pseudo-complement)
  finally show w  $\sqcap$  z  $\sqcap$  xT  $\leq$  bot
    by simp
  thus zT  $\sqcap$  x  $\sqcap$  wT  $\leq$  bot
    by (smt conv-dist-inf conv-inf-bot-iff inf.left-commute
inf.sup-monoid.add-commute le-bot)
  have irreflexive z
    by (meson 4 assms(1) dual-order.trans irreflexive-complement-reflexive
irreflexive-inf-closed reflexive-complement-irreflexive)
  hence asymmetric z
    using 4 by (simp add: pseudo-complement irreflexive-inf-arc-asymmetric)
  thus zT  $\sqcap$  x  $\sqcap$  z  $\sqcap$  xT  $\leq$  bot
    by (simp add: inf.left-commute inf.sup-monoid.add-commute)
qed
finally have acyclic ?y
  using 1 6 by (simp add: le-bot acyclic-2-def)
hence ?y*  $\sqcap$  ?yT = bot
  using acyclic-star-below-complement-1 by blast
hence w*  $\sqcap$  ?yT = bot
  using dual-order.trans pseudo-complement star.circ-sub-dist by blast
hence w*  $\sqcap$  z  $\sqcap$  xT = bot
  by (simp add: comp-inf.semiring.distrib-left conv-dist-inf conv-dist-sup
inf.sup-monoid.add-assoc)
hence z  $\sqcap$  xT = bot
  using 5 by (metis comp-inf.p-pp-comp inf.absorb2 pp-pp-inf-bot-iff)
hence z  $\sqcap$  --x = bot
  using assms(2) pseudo-complement by auto
hence z = bot
  using 4 inf.orderE by auto
thus False
  using 3 4 comp-inf.coreflexive-pseudo-complement inf-bot-right by auto
qed
next
  show acyclic-4 x  $\implies$  acyclic-2 x
    by (simp add: assms(2) acyclic-4-implies-2)
qed

```

end

end

context *kleene-relation-algebra*
begin

Theorem 8

lemma *acyclic-3a-implies-4b*:

assumes *acyclic-3a* x

shows *acyclic-4b* x

proof (*unfold acyclic-4b-def, rule allI, rule impI*)

fix y

let $?y = (x \sqcap -y^*) \sqcup y$

assume $1: y \leq x$

have $x = (x \sqcap -y^*) \sqcup (x \sqcap y^*)$

by *simp*

also have $\dots \leq ?y \sqcup y^*$

using *shunting-var* **by** *fastforce*

also have $\dots \leq ?y^*$

by (*simp add: star.circ-increasing star.circ-sub-dist sup-commute*)

finally have $?y = x$

using 1 *assms acyclic-3a-def* **by** *simp*

hence $x \sqcap y^* = y \sqcap y^*$

by (*smt (z3) inf.sup-monoid.add-commute inf-sup-absorb inf-sup-distrib2 maddux-3-13 sup-commute sup-inf-absorb*)

thus $x \sqcap y^* = y$

by (*simp add: inf-absorb1 star.circ-increasing*)

qed

lemma *acyclic-3a-4b*:

acyclic-3a $x \iff$ *acyclic-4b* x

using *acyclic-3a-implies-4b acyclic-4a-4b acyclic-4a-implies-3a* **by** *blast*

lemma *acyclic-4-4a*:

acyclic-4 $x \iff$ *acyclic-4a* x

by (*simp add: acyclic-4-def acyclic-4a-def*)

2.4 Counterexamples

Calls to *nitpick* have been put into comments to save processing time.

independence of (0)

lemma *symmetric* $x \implies$ *irreflexive-inf* $x \implies$ *orientable* x

nitpick[*expect=genuine,card=4,timeout=600*]

oops

lemma *linear-orderable-6* $x \implies$ *path-orderable* x

```

nitpick[expect=genuine,card=8,timeout=600]
oops
(5) does not imply (6)
lemma symmetric x  $\implies$  irreflexive x  $\implies$  acyclic-5a x  $\implies$  acyclic-6 x
nitpick[expect=genuine,card=4,timeout=600]
oops
(2) does not imply (4)
lemma symmetric x  $\implies$  irreflexive x  $\implies$  acyclic-2 x  $\implies$  acyclic-4 x
nitpick[expect=genuine,card=8,timeout=600]
oops
end
end

```

3 Axioms and Algorithmic Proofs

In this theory we verify the correctness of three basic graph algorithms. We use them to constructively prove a number of graph properties.

```
theory Algorithms
```

```
imports HOL-Hoare.Hoare-Logic Forests
```

```
begin
```

```
context stone-kleene-relation-algebra-arc
```

```
begin
```

Assuming the arc axiom we can define the function *choose-arc* that selects an arc in a non-empty graph.

```
definition choose-arc x  $\equiv$  if x = bot then bot else SOME y . arc y  $\wedge$  y  $\leq$  --x
```

```
lemma choose-arc-below:
```

```
choose-arc x  $\leq$  --x
```

```
proof (cases x = bot)
```

```
case True
```

```
thus ?thesis
```

```
using choose-arc-def by auto
```

```
next
```

```
let ?P =  $\lambda$ y . arc y  $\wedge$  y  $\leq$  --x
```

```
case False
```

```
have ?P (SOME y . ?P y)
```

```
apply (rule someI-ex)
```

```

    using someI-ex False arc-axiom by auto
  thus ?thesis
    using False choose-arc-def by auto
qed

```

```

lemma choose-arc-arc:
  assumes  $x \neq \text{bot}$ 
  shows arc (choose-arc x)
proof -
  let ?P =  $\lambda y . \text{arc } y \wedge y \leq --x$ 
  have ?P (SOME y . ?P y)
    apply (rule someI-ex)
    using someI-ex assms arc-axiom by auto
  thus ?thesis
    using assms choose-arc-def by auto
qed

```

```

lemma choose-arc-bot:
  choose-arc bot = bot
  by (metis bot-unique choose-arc-below regular-closed-bot)

```

```

lemma choose-arc-bot-iff:
  choose-arc x = bot  $\longleftrightarrow$  x = bot
  using covector-bot-closed inf-bot-right choose-arc-arc vector-bot-closed
  choose-arc-bot by fastforce

```

```

lemma choose-arc-regular:
  regular (choose-arc x)
proof (cases x = bot)
  assume x = bot
  thus ?thesis
    by (simp add: choose-arc-bot)
next
  assume  $x \neq \text{bot}$ 
  thus ?thesis
    by (simp add: arc-regular choose-arc-arc)
qed

```

3.1 Constructing a spanning tree

definition *spanning-forest* $f g \equiv \text{forest } f \wedge f \leq --g \wedge \text{components } g \leq \text{forest-components } f \wedge \text{regular } f$

definition *kruskal-spanning-invariant* $f g h \equiv \text{symmetric } g \wedge h = h^T \wedge g \sqcap --h = h \wedge \text{spanning-forest } f (-h \sqcap g)$

```

lemma spanning-forest-spanning:
  spanning-forest f g  $\implies$  spanning (--g) f
  by (smt (z3) cancel-separate-1 order-trans star.circ-increasing
  spanning-forest-def)

```

lemma *spanning-forest-spanning-regular*:
assumes *regular f*
and *regular g*
shows *spanning-forest f g \longleftrightarrow spanning g f*
by (*smt (z3) assms cancel-separate-1 components-increasing dual-order.trans forest-components-star star-isotone spanning-forest-def*)

We prove total correctness of Kruskal's spanning tree algorithm (ignoring edge weights) [6]. The algorithm and proof are adapted from the AFP theory *Relational-Minimum-Spanning-Trees.Kruskal* to work in Stone-Kleene relation algebras [3, 4].

lemma *kruskal-vc-1*:
assumes *symmetric g*
shows *kruskal-spanning-invariant bot g g*
proof (*unfold kruskal-spanning-invariant-def, intro conjI*)
show *symmetric g*
using *assms* **by** *simp*
next
show *g = g^T*
using *assms* **by** *simp*
next
show *g \sqcap --g = g*
using *inf.sup-monoid.add-commute selection-closed-id* **by** *simp*
next
show *spanning-forest bot (-g \sqcap g)*
using *star.circ-transitive-equal spanning-forest-def* **by** *simp*
qed

For the remainder of this theory we assume there are finitely many regular elements. This means that the graphs are finite and is needed for proving termination of the algorithms.

context
assumes *finite-regular: finite { x . regular x }*
begin

lemma *kruskal-vc-2*:
assumes *kruskal-spanning-invariant f g h*
and *h \neq bot*
shows (*choose-arc h \leq -forest-components f \longrightarrow kruskal-spanning-invariant*
*((f \sqcap -(top * choose-arc h * f^{T*})) \sqcup (f \sqcap top * choose-arc h * f^{T*})^T \sqcup*
choose-arc h) g (h \sqcap -choose-arc h \sqcap -choose-arc h^T)
 \wedge *card { x . regular x \wedge x \leq --h \wedge x \leq*
 $\text{-choose-arc h \wedge x \leq -choose-arc h^T } < card { x . regular x \wedge x \leq --h }) \wedge
(\neg choose-arc h \leq -forest-components f \longrightarrow kruskal-spanning-invariant f
g (h \sqcap -choose-arc h \sqcap -choose-arc h^T)
 \wedge *card { x . regular x \wedge x \leq --h \wedge x \leq*
 $\text{-choose-arc h \wedge x \leq -choose-arc h^T } < card { x . regular x \wedge x \leq --h })
proof -$$

```

let ?e = choose-arc h
let ?f = (f  $\sqcap$   $\neg$ (top * ?e * fT*))  $\sqcup$  (f  $\sqcap$  top * ?e * fT*)T  $\sqcup$  ?e
let ?h = h  $\sqcap$   $\neg$ ?e  $\sqcap$   $\neg$ ?eT
let ?F = forest-components f
let ?n1 = card { x . regular x  $\wedge$  x  $\leq$   $\neg\neg$ h }
let ?n2 = card { x . regular x  $\wedge$  x  $\leq$   $\neg\neg$ h  $\wedge$  x  $\leq$   $\neg$ ?e  $\wedge$  x  $\leq$   $\neg$ ?eT }
have 1: regular f  $\wedge$  regular ?e
  by (metis assms(1) kruskal-spanning-invariant-def spanning-forest-def
choose-arc-regular)
hence 2: regular ?f  $\wedge$  regular ?F  $\wedge$  regular (?eT)
  using regular-closed-star regular-conv-closed regular-mult-closed by simp
have 3:  $\neg$  ?e  $\leq$   $\neg$ ?e
  using assms(2) inf.orderE choose-arc-bot-iff by fastforce
have 4: ?n2 < ?n1
  apply (rule psubset-card-mono)
  using finite-regular apply simp
  using 1 3 kruskal-spanning-invariant-def choose-arc-below by auto
show (?e  $\leq$   $\neg$ ?F  $\longrightarrow$  kruskal-spanning-invariant ?f g ?h  $\wedge$  ?n2 < ?n1)  $\wedge$  ( $\neg$  ?e
 $\leq$   $\neg$ ?F  $\longrightarrow$  kruskal-spanning-invariant f g ?h  $\wedge$  ?n2 < ?n1)
proof (rule conjI)
  have 5: injective ?f
  apply (rule kruskal-injective-inv)
  using assms(1) kruskal-spanning-invariant-def spanning-forest-def apply
simp
  apply (simp add: covector-mult-closed)
  apply (simp add: comp-associative comp-isotone star.right-plus-below-circ)
  apply (meson mult-left-isotone order-lesseq-imp star-outer-increasing
top.extremum)
  using assms(1,2) kruskal-spanning-invariant-def kruskal-injective-inv-2
choose-arc-arc spanning-forest-def apply simp
  using assms(2) arc-injective choose-arc-arc apply blast
  using assms(1,2) kruskal-spanning-invariant-def kruskal-injective-inv-3
choose-arc-arc spanning-forest-def by simp
show ?e  $\leq$   $\neg$ ?F  $\longrightarrow$  kruskal-spanning-invariant ?f g ?h  $\wedge$  ?n2 < ?n1
proof
  assume 6: ?e  $\leq$   $\neg$ ?F
  have 7: equivalence ?F
  using assms(1) kruskal-spanning-invariant-def
forest-components-equivalence spanning-forest-def by simp
  have ?eT * top * ?eT = ?eT
  using assms(2) by (simp add: arc-top-arc choose-arc-arc)
  hence ?eT * top * ?eT  $\leq$   $\neg$ ?F
  using 6 7 conv-complement conv-isotone by fastforce
  hence 8: ?e * ?F * ?e = bot
  using le-bot triple-schroeder-p by simp
show kruskal-spanning-invariant ?f g ?h  $\wedge$  ?n2 < ?n1
proof (unfold kruskal-spanning-invariant-def, intro conjI)
  show symmetric g
  using assms(1) kruskal-spanning-invariant-def by simp

```

```

next
  show  $?h = ?h^T$ 
    using assms(1) by (simp add: conv-complement conv-dist-inf
inf-commute inf-left-commute kruskal-spanning-invariant-def)
next
  show  $g \sqcap --?h = ?h$ 
    using 1 2 by (metis assms(1) kruskal-spanning-invariant-def inf-assoc
pp-dist-inf)
next
  show spanning-forest ?f ( $-?h \sqcap g$ )
  proof (unfold spanning-forest-def, intro conjI)
    show injective ?f
      using 5 by simp
next
  show acyclic ?f
    apply (rule kruskal-acyclic-inv)
    using assms(1) kruskal-spanning-invariant-def spanning-forest-def
apply simp
  apply (simp add: covector-mult-closed)
  using 8 assms(1) kruskal-spanning-invariant-def spanning-forest-def
kruskal-acyclic-inv-1 apply simp
  using 8 apply (metis comp-associative mult-left-sub-dist-sup-left
star.circ-loop-fixpoint sup-commute le-bot)
  using 6 by (simp add: p-antitone-iff)
next
  show  $?f \leq --(-?h \sqcap g)$ 
    apply (rule kruskal-subgraph-inv)
    using assms(1) kruskal-spanning-invariant-def spanning-forest-def
apply simp
  using assms(1) apply (metis kruskal-spanning-invariant-def
choose-arc-below order.trans pp-isotone-inf)
  using assms(1) kruskal-spanning-invariant-def apply simp
  using assms(1) kruskal-spanning-invariant-def by simp
next
  show components ( $-?h \sqcap g$ )  $\leq$  forest-components ?f
    apply (rule kruskal-spanning-inv)
    using 5 apply simp
    using 1 regular-closed-star regular-conv-closed regular-mult-closed apply
simp
    using 1 apply simp
    using assms(1) kruskal-spanning-invariant-def spanning-forest-def by
simp
next
  show regular ?f
    using 2 by simp
qed
next
  show  $?n2 < ?n1$ 
    using 4 by simp

```

```

    qed
  qed
next
show  $\neg ?e \leq -?F \longrightarrow$  kruskal-spanning-invariant  $f g ?h \wedge ?n2 < ?n1$ 
proof
  assume  $\neg ?e \leq -?F$ 
  hence  $g: ?e \leq ?F$ 
    using 2 assms(2) arc-in-partition choose-arc-arc by fastforce
  show kruskal-spanning-invariant  $f g ?h \wedge ?n2 < ?n1$ 
  proof (unfold kruskal-spanning-invariant-def, intro conjI)
    show symmetric  $g$ 
      using assms(1) kruskal-spanning-invariant-def by simp
  next
  show  $?h = ?h^T$ 
    using assms(1) by (simp add: conv-complement conv-dist-inf
inf-commute inf-left-commute kruskal-spanning-invariant-def)
  next
  show  $g \sqcap --?h = ?h$ 
    using 1 2 by (metis assms(1) kruskal-spanning-invariant-def inf-assoc
pp-dist-inf)
  next
  show spanning-forest  $f (-?h \sqcap g)$ 
  proof (unfold spanning-forest-def, intro conjI)
    show injective  $f$ 
      using assms(1) kruskal-spanning-invariant-def spanning-forest-def by
simp
  next
  show acyclic  $f$ 
    using assms(1) kruskal-spanning-invariant-def spanning-forest-def by
simp
  next
  have  $f \leq --(-h \sqcap g)$ 
    using assms(1) kruskal-spanning-invariant-def spanning-forest-def by
simp
  also have  $\dots \leq --(-?h \sqcap g)$ 
    using comp-inf.mult-right-isotone inf.sup-monoid.add-commute
inf-left-commute p-antitone-inf pp-isotone by auto
  finally show  $f \leq --(-?h \sqcap g)$ 
    by simp
  next
  show components  $(-?h \sqcap g) \leq ?F$ 
    apply (rule kruskal-spanning-inv-1)
    using 9 apply simp
    using 1 apply simp
    using assms(1) kruskal-spanning-invariant-def spanning-forest-def
apply simp
    using assms(1) kruskal-spanning-invariant-def
forest-components-equivalence spanning-forest-def by simp
  next

```



```

    show regular f
      using 1 by simp
    qed
  next
    show ?n2 < ?n1
      using 4 by simp
    qed
  qed
qed
qed

```

theorem *kruskal-spanning*:

```

  VARS e f h
  [ symmetric g ]
  f := bot;
  h := g;
  WHILE h ≠ bot
    INV { kruskal-spanning-invariant f g h }
    VAR { card { x . regular x ∧ x ≤ --h } }
    DO e := choose-arc h;
      IF e ≤ -forest-components f THEN
        f := (f □ -(top * e * fT*)) □ (f □ top * e * fT*)T □ e
      ELSE
        SKIP
      FI;
      h := h □ -e □ -eT
    OD
  [ spanning-forest f g ]
  apply vcg-tc-simp
  using kruskal-vc-1 apply simp
  using kruskal-vc-2 apply simp
  using kruskal-spanning-invariant-def by auto

```

lemma *kruskal-exists-spanning*:

```

  symmetric g ⇒ ∃ f . spanning-forest f g
  using tc-extract-function kruskal-spanning by blast

```

Theorem 16

lemma *symmetric-spannable*:

```

  symmetric g ⇒ spannable (--g)
  using kruskal-exists-spanning spanning-forest-spanning spannable-def by blast

```

3.2 Breadth-first search

We prove total correctness of a simple breadth-first search algorithm. It is a variant of an algorithm discussed in [1].

theorem *bfs-reachability*:

```

  VARS p q t
  [ regular r ∧ regular s ∧ vector s ]

```

```

t := bot;
q := s;
p := -s  $\sqcap$  rT * s;
WHILE p ≠ bot
  INV { regular r  $\wedge$  regular q  $\wedge$  vector q  $\wedge$  asymmetric t  $\wedge$  t ≤ r  $\wedge$  t ≤ q  $\wedge$  q =
tT* * s  $\wedge$  p = -q  $\sqcap$  rT * q }
  VAR { card { x . regular x  $\wedge$  x ≤ --(-q  $\sqcap$  rT* * s) } }
  DO t := t  $\sqcup$  (r  $\sqcap$  q * pT);
    q := q  $\sqcup$  p;
    p := -q  $\sqcap$  rT * p
  OD
[ asymmetric t  $\wedge$  t ≤ r  $\wedge$  q = tT* * s  $\wedge$  q = rT* * s ]
proof veg-tc
  fix p q t
  assume regular r  $\wedge$  regular s  $\wedge$  vector s
  thus regular r  $\wedge$  regular s  $\wedge$  vector s  $\wedge$  asymmetric bot  $\wedge$  bot ≤ r  $\wedge$  bot ≤ s  $\wedge$  s
= botT* * s  $\wedge$  -s  $\sqcap$  rT * s = -s  $\sqcap$  rT * s
  by (simp add: star.circ-zero)
next
  fix p q t
  assume 1: (regular r  $\wedge$  regular q  $\wedge$  vector q  $\wedge$  asymmetric t  $\wedge$  t ≤ r  $\wedge$  t ≤ q  $\wedge$ 
q = tT* * s  $\wedge$  p = -q  $\sqcap$  rT * q)  $\wedge$  ¬ p ≠ bot
  have q = rT* * s
  apply (rule order.antisym)
  using 1 conv-order mult-left-isotone star-isotone apply simp
  using 1 by (metis inf.sup-monoid.add-commute mult-1-left mult-left-isotone
mult-right-isotone order-lesseq-imp pseudo-complement star.circ-reflexive
star-left-induct-mult)
  thus asymmetric t  $\wedge$  t ≤ r  $\wedge$  q = tT* * s  $\wedge$  q = rT* * s
  using 1 by simp
next
  fix n p q t
  assume 2: ((regular r  $\wedge$  regular q  $\wedge$  vector q  $\wedge$  asymmetric t  $\wedge$  t ≤ r  $\wedge$  t ≤ q  $\wedge$ 
q = tT* * s  $\wedge$  p = -q  $\sqcap$  rT * q)  $\wedge$  p ≠ bot)  $\wedge$  card { x . regular x  $\wedge$  x ≤ --(-q
 $\sqcap$  rT* * s) } = n
  hence 3: vector p
  using vector-complement-closed vector-inf-closed vector-mult-closed by blast
  show -(q  $\sqcup$  p)  $\sqcap$  rT * p, q  $\sqcup$  p, t  $\sqcup$  (r  $\sqcap$  q * pT)
  ∈ { trip . (case trip of (p, q, t) ⇒ regular r  $\wedge$  regular q  $\wedge$  vector q  $\wedge$ 
asymmetric t  $\wedge$  t ≤ r  $\wedge$  t ≤ q  $\wedge$  q = tT* * s  $\wedge$  p = -q  $\sqcap$  rT * q)  $\wedge$ 
(case trip of (p, q, t) ⇒ card { x . regular x  $\wedge$  x ≤ --(-q  $\sqcap$  rT*
* s) } ) < n }
  apply (rule CollectI, rule conjI)
  subgoal proof (intro case-prodI, intro conjI)
  show regular r
  using 2 by blast
  show regular (q  $\sqcup$  p)
  using 2 regular-conv-closed regular-mult-closed by force
  show vector (q  $\sqcup$  p)

```

using 2 *vector-complement-closed vector-inf-closed vector-mult-closed vector-sup-closed* **by force**
show *asymmetric* $(t \sqcup (r \sqcap q * p^T))$
proof –
have $t \sqcap (r \sqcap q * p^T)^T \leq t \sqcap p * q^T$
by (*metis comp-inf.mult-right-isotone conv-dist-comp conv-involutive conv-order inf.cobounded2*)
also have $\dots \leq t \sqcap p$
using 3 **by** (*metis comp-inf.mult-right-isotone comp-inf.star.circ-sup-sub-sup-one-1 inf.boundedE le-sup-iff mult-right-isotone*)
finally have 4: $t \sqcap (r \sqcap q * p^T)^T = \text{bot}$
using 2 **by** (*metis order.antisym bot-least inf.sup-monoid.add-assoc pseudo-complement*)
hence 5: $r \sqcap q * p^T \sqcap t^T = \text{bot}$
using *conv-inf-bot-iff inf-commute* **by force**
have $r \sqcap q * p^T \sqcap (r \sqcap q * p^T)^T \leq q * p^T \sqcap p * q^T$
by (*metis comp-inf.comp-isotone conv-dist-comp conv-involutive conv-order inf.cobounded2*)
also have $\dots \leq q \sqcap p$
using 2 3 **by** (*metis comp-inf.comp-isotone inf.cobounded1 vector-covector*)
finally have 6: $r \sqcap q * p^T \sqcap (r \sqcap q * p^T)^T = \text{bot}$
using 2 **by** (*metis inf.cobounded1 inf.sup-monoid.add-commute le-bot pseudo-complement*)
have $(t \sqcup (r \sqcap q * p^T)) \sqcap (t \sqcup (r \sqcap q * p^T))^T = (t \sqcap t^T) \sqcup (t \sqcap (r \sqcap q * p^T)^T) \sqcup (r \sqcap q * p^T \sqcap t^T) \sqcup (r \sqcap q * p^T \sqcap (r \sqcap q * p^T)^T)$
by (*simp add: sup commute sup.left-commute conv-dist-sup inf-sup-distrib1 inf-sup-distrib2*)
also have $\dots = \text{bot}$
using 2 4 5 6 **by auto**
finally show ?thesis
.

qed
show $t \sqcup (r \sqcap q * p^T) \leq r$
using 2 **by** (*meson inf.cobounded1 le-supI*)
show $t \sqcup (r \sqcap q * p^T) \leq q \sqcup p$
using 2 **by** (*metis comp-inf.star.circ-sup-sub-sup-one-1 inf.absorb2 inf.coboundedI2 inf.sup-monoid.add-commute le-sup-iff mult-right-isotone sup-left-divisibility*)
show $q \sqcup p = (t \sqcup (r \sqcap q * p^T))^{T*} * s$
proof (*rule order.antisym*)
have 7: $q \leq (t \sqcup (r \sqcap q * p^T))^{T*} * s$
using 2 **by** (*metis conv-order mult-left-isotone star-isotone sup-left-divisibility*)
have $-q \sqcap (r \sqcap q * p^T)^T * q \leq (t \sqcup (r \sqcap q * p^T))^T * t^{T*} * s$
using 2 *comp-associative conv-dist-sup inf.coboundedI2*
mult-right-sub-dist-sup-right **by auto**
also have $\dots \leq (t \sqcup (r \sqcap q * p^T))^{T*} * s$
by (*simp add: conv-dist-sup mult-left-isotone star.circ-increasing*)

star.circ-mult-upper-bound star.circ-sub-dist
finally have 8: $-q \sqcap (r \sqcap q * p^T)^T * q \leq (t \sqcup (r \sqcap q * p^T))^{T*} * s$
 \cdot
have 9: $(r \sqcap -q)^T * q = \text{bot}$
using 2 by (*metis conv-dist-inf covector-inf-comp-3 pp-inf-p semiring.mult-not-zero vector-complement-closed*)
have $-q \sqcap (r \sqcap -(q * p^T))^T * q = -q \sqcap (r \sqcap (-q \sqcup -p^T))^T * q$
using 2 3 by (*metis p-dist-inf vector-covector*)
also have $\dots = (-q \sqcap (r \sqcap -q)^T * q) \sqcup (-q \sqcap (r \sqcap -p^T)^T * q)$
by (*simp add: conv-dist-sup inf-sup-distrib1 mult-right-dist-sup*)
also have $\dots = -q \sqcap (r \sqcap -p^T)^T * q$
using 9 by *simp*
also have $\dots = -q \sqcap -p \sqcap r^T * q$
using 3 by (*metis conv-complement conv-dist-inf conv-involutive inf.sup-monoid.add-assoc inf-vector-comp vector-complement-closed*)
finally have 10: $-q \sqcap (r \sqcap -(q * p^T))^T * q = \text{bot}$
using 2 *inf-import-p pseudo-complement* **by** *auto*
have $r = (r \sqcap q * p^T) \sqcup (r \sqcap -(q * p^T))$
using 2 by (*smt (z3) maddux-3-11-pp pp-dist-comp regular-closed-inf regular-conv-closed*)
hence $p = -q \sqcap ((r \sqcap q * p^T) \sqcup (r \sqcap -(q * p^T)))^T * q$
using 2 by *auto*
also have $\dots = (-q \sqcap (r \sqcap q * p^T)^T * q) \sqcup (-q \sqcap (r \sqcap -(q * p^T))^T * q)$
by (*simp add: conv-dist-sup inf-sup-distrib1 semiring.distrib-right*)
also have $\dots \leq (t \sqcup (r \sqcap q * p^T))^{T*} * s$
using 8 10 *le-sup-iff bot-least* **by** *blast*
finally show $q \sqcup p \leq (t \sqcup (r \sqcap q * p^T))^{T*} * s$
using 7 by *simp*
have 11: $t^T * q \leq r^T * q$
using 2 *conv-order mult-left-isotone* **by** *auto*
have $t^T * p \leq t^T * \text{top}$
by (*simp add: mult-right-isotone*)
also have $\dots = t^T * q \sqcup t^T * -q$
using 2 *regular-complement-top semiring.distrib-left* **by** *force*
also have $\dots = t^T * q$
proof $-$
have $t^T * -q = \text{bot}$
using 2 by (*metis bot-least conv-complement-sub conv-dist-comp conv-involutive mult-right-isotone regular-closed-bot stone sup.absorb2 sup-commute*)
thus *?thesis*
by *simp*
qed
finally have 12: $t^T * p \leq r^T * q$
using 11 *order.trans* **by** *blast*
have 13: $(r \sqcap q * p^T)^T * q \leq r^T * q$
by (*simp add: conv-dist-inf mult-left-isotone*)
have $(r \sqcap q * p^T)^T * p \leq p$
using 3 by (*metis conv-dist-comp conv-dist-inf conv-involutive*)

inf.coboundedI2 mult-isotone mult-right-isotone top.extremum
hence 14: $(r \sqcap q * p^T)^T * p \leq r^T * q$
using 2 le-infE by blast
have $(t \sqcup (r \sqcap q * p^T))^T * (q \sqcup p) = t^T * q \sqcup t^T * p \sqcup (r \sqcap q * p^T)^T * q$
 $\sqcup (r \sqcap q * p^T)^T * p$
by (*metis conv-dist-sup semiring.distrib-left semiring.distrib-right sup-assoc*)
also have $\dots \leq r^T * q$
using 11 12 13 14 by simp
finally have $(t \sqcup (r \sqcap q * p^T))^T * (q \sqcup p) \leq q \sqcup p$
using 2 by (*metis maddux-3-21-pp sup.boundedE sup-right-divisibility*)
thus $(t \sqcup (r \sqcap q * p^T))^T * s \leq q \sqcup p$
using 2 by (*smt (verit, ccfv-SIG) star.circ-loop-fixpoint star-left-induct sup.bounded-iff sup-left-divisibility*)
qed
show $-(q \sqcup p) \sqcap r^T * p = -(q \sqcup p) \sqcap r^T * (q \sqcup p)$
proof (*rule order.antisym*)
show $-(q \sqcup p) \sqcap r^T * p \leq -(q \sqcup p) \sqcap r^T * (q \sqcup p)$
using inf.sup-right-isotone mult-left-sub-dist-sup-right by blast
have 15: $-(q \sqcup p) \sqcap r^T * (q \sqcup p) = -(q \sqcup p) \sqcap r^T * q \sqcup -(q \sqcup p) \sqcap r^T * p$
by (*simp add: comp-inf.semiring.distrib-left mult-left-dist-sup*)
have $-(q \sqcup p) \sqcap r^T * q \leq -(q \sqcup p) \sqcap r^T * p$
using 2 by (*metis bot-least p-dist-inf p-dist-sup p-inf-sup-below pseudo-complement*)
thus $-(q \sqcup p) \sqcap r^T * (q \sqcup p) \leq -(q \sqcup p) \sqcap r^T * p$
using 15 sup.absorb2 by force
qed
qed
subgoal proof clarsimp
have $\text{card } \{ x . \text{regular } x \wedge x \leq -q \wedge x \leq -p \wedge x \leq \neg\neg(r^{T*} * s) \} < \text{card } \{ x . \text{regular } x \wedge x \leq \neg\neg(-q \sqcap r^{T*} * s) \}$
proof (*rule psubset-card-mono*)
show $\text{finite } \{ x . \text{regular } x \wedge x \leq \neg\neg(-q \sqcap r^{T*} * s) \}$
using finite-regular by simp
show $\{ x . \text{regular } x \wedge x \leq -q \wedge x \leq -p \wedge x \leq \neg\neg(r^{T*} * s) \} \subset \{ x . \text{regular } x \wedge x \leq \neg\neg(-q \sqcap r^{T*} * s) \}$
proof $-$
have $\forall x . x \leq -q \wedge x \leq \neg\neg(r^{T*} * s) \longrightarrow x \leq \neg\neg(-q \sqcap r^{T*} * s)$
by auto
hence 16: $\{ x . \text{regular } x \wedge x \leq -q \wedge x \leq -p \wedge x \leq \neg\neg(r^{T*} * s) \} \subseteq \{ x . \text{regular } x \wedge x \leq \neg\neg(-q \sqcap r^{T*} * s) \}$
by blast
have 17: *regular p*
using 2 regular-conv-closed regular-mult-closed by force
hence 18: $\neg p \leq -p$
using 2 by (*metis inf.absorb1 pp-inf-p*)
have 19: $p \leq -q$
using 2 by simp

have $r^T * q \leq r^{T*} * s$
using 2 **by** (*metis (no-types, lifting) comp-associative conv-dist-sup mult-left-isotone star.circ-increasing star.circ-mult-upper-bound star.circ-sub-dist sup-left-divisibility*)
hence 20: $p \leq --(r^{T*} * s)$
using 2 *le-infI2 order-lesseq-imp pp-increasing* **by** *blast*
hence 21: $p \leq --(-q \sqcap r^{T*} * s)$
using 2 **by** *simp*
show *?thesis*
using 16 17 18 19 20 21 **by** *blast*
qed
qed
thus $\text{card } \{ x . \text{regular } x \wedge x \leq -q \wedge x \leq -p \wedge x \leq --(r^{T*} * s) \} < n$
using 2 **by** *auto*
qed
done
qed

Theorem 18

lemma *bfs-reachability-exists*:

$\text{regular } r \wedge \text{regular } s \wedge \text{vector } s \implies \exists t . \text{asymmetric } t \wedge t \leq r \wedge t^{T*} * s = r^{T*} * s$
using *tc-extract-function bfs-reachability* **by** *blast*

Theorem 17

lemma *orientable-path*:

$\text{arc } x \implies x \leq --y^* \implies \exists z . z \leq y \wedge \text{asymmetric } z \wedge x \leq --z^*$
proof –

assume 1: *arc x* **and** 2: $x \leq --y^*$
hence $\text{regular } (-y) \wedge \text{regular } (x * \text{top}) \wedge \text{vector } (x * \text{top})$
using *bijjective-regular mult-assoc* **by** *auto*
from this obtain t **where** 3: $\text{asymmetric } t \wedge t \leq --y \wedge t^{T*} * (x * \text{top}) = (-y)^{T*} * (x * \text{top})$
using *bfs-reachability-exists* **by** *blast*
let $?z = t \sqcap y$
have $x^T * \text{top} * x^T \leq (-y)^{T*}$
using 1 2 **by** (*metis arc-top-arc conv-complement conv-isotone conv-star-commute arc-conv-closed pp-dist-star*)
hence $x^T \leq (-y)^{T*} * x * \text{top}$
using 1 *comp-associative conv-dist-comp shunt-bijjective* **by** *force*
also have $\dots = t^{T*} * x * \text{top}$
using 3 *mult-assoc* **by** *force*
finally have $x^T * \text{top} * x^T \leq t^{T*}$
using 1 *comp-associative conv-dist-comp shunt-bijjective* **by** *force*
hence $x^T \leq t^{T*}$
using 1 **by** (*metis arc-top-arc arc-conv-closed*)
also have $\dots \leq (--?z)^{T*}$
using 3 **by** (*metis conv-order inf.orderE inf-pp-semi-commute star-isotone*)
finally have $x \leq --?z^*$

```

    using conv-order conv-star-commute pp-dist-star by fastforce
  thus  $\exists z . z \leq y \wedge \text{asymmetric } z \wedge x \leq \neg\neg z^*$ 
    using  $\exists$  asymmetric-inf-closed inf.cobounded2 by blast
qed

```

3.3 Extending partial orders to linear orders

We prove total correctness of Szpilrajn's algorithm [7]. A partial-correctness proof using Prover9 is given in [2].

theorem *szpilrajn*:

```

  VARS e t
  [ order p  $\wedge$  regular p ]
  t := p;
  WHILE t  $\sqcup$  tT  $\neq$  top
    INV { order t  $\wedge$  regular t  $\wedge$  p  $\leq$  t }
    VAR { card { x . regular x  $\wedge$  x  $\leq$   $\neg$ (t  $\sqcup$  tT) } }
    DO e := choose-arc ( $\neg$ (t  $\sqcup$  tT));
      t := t  $\sqcup$  t * e * t
    OD
  [ linear-order t  $\wedge$  p  $\leq$  t ]

```

proof *vcg-tc-simp*

```

  fix t
  let ?e = choose-arc ( $\neg$ t  $\sqcap$   $\neg$ tT)
  let ?tet = t * ?e * t
  let ?t = t  $\sqcup$  ?tet
  let ?s1 = { x . regular x  $\wedge$  x  $\leq$   $\neg$ t  $\wedge$  x  $\leq$   $\neg$ ?tet  $\wedge$  x  $\leq$   $\neg$ tT }
  let ?s2 = { x . regular x  $\wedge$  x  $\leq$   $\neg$ t  $\wedge$  x  $\leq$   $\neg$ tT }
  assume 1: reflexive t  $\wedge$  transitive t  $\wedge$  antisymmetric t  $\wedge$  regular t  $\wedge$  p  $\leq$  t  $\wedge$   $\neg$ 
linear t

```

```

  show reflexive ?t  $\wedge$ 
    transitive ?t  $\wedge$ 
    antisymmetric ?t  $\wedge$ 
    ?t = t  $\sqcup$   $\neg$ ?tet  $\wedge$ 
    p  $\leq$  ?t  $\wedge$ 
    card ?s1 < card ?s2

```

proof (*intro conjI*)

```

  show reflexive ?t
    using 1 by (simp add: sup.coboundedI1)
  have  $\neg$ t  $\sqcap$   $\neg$ tT  $\neq$  bot
    using 1 regular-closed-top regular-conv-closed by force
  hence 2: arc ?e
    using choose-arc-arc by blast
  have ?t * ?t = t * t  $\sqcup$  t * ?tet  $\sqcup$  ?tet * t  $\sqcup$  ?tet * ?tet
    by (smt (z3) mult-left-dist-sup mult-right-dist-sup sup-assoc)
  also have ...  $\leq$  ?t
  proof (intro sup-least)
    show t * t  $\leq$  ?t
      using 1 sup.coboundedI1 by blast
    show t * ?tet  $\leq$  ?t

```

```

    using 1 by (metis le-supI2 mult-left-isotone mult-assoc)
  show ?tet * t ≤ ?t
    using 1 mult-right-isotone sup.coboundedI2 mult-assoc by auto
  have ?e * t * t * ?e ≤ ?e
    using 2 by (smt arc-top-arc mult-assoc mult-right-isotone mult-left-isotone
top-greatest)
  hence transitive ?tet
    by (smt mult-assoc mult-right-isotone mult-left-isotone)
  thus ?tet * ?tet ≤ ?t
    using le-supI2 by auto
qed
finally show transitive ?t
.
have 3: ?e ≤ -tT
  by (metis choose-arc-below inf.cobounded2 order-lesseq-imp p-dist-sup
regular-closed-p)
have 4: ?e ≤ -t
  by (metis choose-arc-below inf.cobounded1 order-trans regular-closed-inf
regular-closed-p)
have ?t ∩ ?tT = (t ∩ tT) ∪ (t ∩ ?tetT) ∪ (?tet ∩ tT) ∪ (?tet ∩ ?tetT)
  by (smt (z3) conv-dist-sup inf-sup-distrib1 inf-sup-distrib2
sup-monoid.add-assoc)
also have ... ≤ 1
proof (intro sup-least)
  show antisymmetric t
    using 1 by simp
  have t * t * t = t
    using 1 preorder-idempotent by fastforce
  also have ... ≤ -?eT
    using 3 by (metis p-antitone-iff conv-complement conv-order
conv-involutive)
  finally have tT * ?eT * tT ≤ -t
    using triple-schroeder-p by blast
  hence t ∩ ?tetT = bot
    by (simp add: comp-associative conv-dist-comp p-antitone
pseudo-complement-pp)
  thus t ∩ ?tetT ≤ 1
    by simp
  thus ?tet ∩ tT ≤ 1
    by (smt conv-isotone inf-commute conv-one conv-dist-inf conv-involutive)
  have ?e * t * ?e ≤ ?e
    using 2 by (smt arc-top-arc mult-assoc mult-right-isotone mult-left-isotone
top-greatest)
  also have ... ≤ -tT
    using 3 by simp
  finally have ?tet ≤ -?eT
    by (metis conv-dist-comp schroeder-3-p triple-schroeder-p)
  hence t * t * ?e * t * t ≤ -?eT
    using 1 by (metis preorder-idempotent mult-assoc)

```


hence $t^T * ?e^T * t^T \leq -?tet$
using *triple-schroeder-p mult-assoc by auto*
hence $?tet \sqcap ?tet^T = \text{bot}$
by (*simp add: conv-dist-comp p-antitone pseudo-complement-pp mult-assoc*)
thus *antisymmetric ?tet*
by *simp*
qed
finally show *antisymmetric ?t*
.
show $?t = t \sqcup --?tet$
using *1 choose-arc-regular regular-mult-closed by auto*
show $p \leq ?t$
using *1 by (simp add: le-supI1)*
show $\text{card } ?s1 < \text{card } ?s2$
proof (*rule psubset-card-mono*)
show *finite* $\{ x . \text{regular } x \wedge x \leq -t \wedge x \leq -t^T \}$
using *finite-regular by simp*
show $\{ x . \text{regular } x \wedge x \leq -t \wedge x \leq -?tet \wedge x \leq -?t^T \} \subset \{ x . \text{regular } x$
 $\wedge x \leq -t \wedge x \leq -t^T \}$
proof –
have $\forall x . \text{regular } x \wedge x \leq -t \wedge x \leq -?tet \wedge x \leq -?t^T \longrightarrow \text{regular } x \wedge x$
 $\leq -t \wedge x \leq -t^T$
using *conv-dist-sup by auto*
hence *5*: $\{ x . \text{regular } x \wedge x \leq -t \wedge x \leq -?tet \wedge x \leq -?t^T \} \subseteq \{ x .$
regular $x \wedge x \leq -t \wedge x \leq -t^T \}$
by *blast*
have *6*: *regular* $?e \wedge ?e \leq -t \wedge ?e \leq -t^T$
using *2 3 4 choose-arc-regular by blast*
have $\neg ?e \leq -?tet$
proof
assume *7*: $?e \leq -?tet$
have $?e \leq ?e * t$
using *1 by (meson mult-right-isotone mult-sub-right-one order.trans)*
also have $?e * t \leq -(t^T * ?e)$
using *7 p-antitone-iff schroeder-3-p mult-assoc by auto*
also have $\dots \leq -(t^T * ?e)$
using *1 conv-isotone mult-left-isotone p-antitone by blast*
also have $\dots = -?e$
by *simp*
finally show *False*
using *1 2 by (smt (z3) bot-least eq-refl inf.absorb1 pseudo-complement*
semiring.mult-not-zero top-le)
qed
thus *?thesis*
using *5 6 by blast*
qed
qed
qed
qed

Theorem 15

lemma *szpilrajn-exists*:

order $p \wedge$ *regular* $p \implies \exists t .$ *linear-order* $t \wedge p \leq t$

using *tc-extract-function szpilrajn* **by** *blast*

lemma *complement-one-transitively-orientable*:

transitively-orientable (-1)

proof –

have $\exists t .$ *linear-order* t

using *szpilrajn-exists* *bijjective-one-closed* *bijjective-regular* *order-one-closed* **by**

blast

thus *?thesis*

using *exists-split-characterisations(4)* **by** *blast*

qed

end

end

end

References

- [1] R. Berghammer. Combining relational calculus and the Dijkstra–Gries method for deriving relational programs. *Information Sciences*, 119(3–4):155–171, 1999.
- [2] R. Berghammer and G. Struth. On automated program construction and verification. In C. Bolduc, J. Desharnais, and B. Ktari, editors, *Mathematics of Program Construction*, volume 6120 of *Lecture Notes in Computer Science*, pages 22–41. Springer, 2010.
- [3] W. Guttman. Stone-Kleene relation algebras. *Archive of Formal Proofs*, 2017.
- [4] W. Guttman. Verifying minimum spanning tree algorithms with Stone relation algebras. *Journal of Logical and Algebraic Methods in Programming*, 101:132–150, 2018.
- [5] W. Guttman. Second-order properties of undirected graphs. In U. Fahrenberg, M. Gehrke, L. Santocanale, and M. Winter, editors, *Relational and Algebraic Methods in Computer Science (RAMiCS 2021)*, Lecture Notes in Computer Science. Springer, 2021. To appear.
- [6] J. B. Kruskal, Jr. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7(1):48–50, 1956.

- [7] E. Szpilrajn. Sur l'extension de l'ordre partiel. *Fundamenta Mathematicae*, 16:386–389, 1930.