Stone-Kleene Relation Algebras

Walter Guttmann

October 13, 2025

Abstract

We develop Stone-Kleene relation algebras, which expand Stone relation algebras with a Kleene star operation to describe reachability in weighted graphs. Many properties of the Kleene star arise as a special case of a more general theory of iteration based on Conway semirings extended by simulation axioms. This includes several theorems representing complex program transformations. We formally prove the correctness of Conway's automata-based construction of the Kleene star of a matrix. We prove numerous results useful for reasoning about weighted graphs.

Contents

1 Synopsis and Motivation

This document describes the following five theory files:

* Iterings describes a general iteration operation that works for many different computation models. We first consider equational axioms based on variants of Conway semirings. We expand these structures by generalised simulation axioms, which hold in total and general correctness models, not just in partial correctness models like the induction axioms. Simulation axioms are still powerful enough to prove separation theorems and Back's atomicity refinement theorem [4].

* Kleene Algebras form a particular instance of iterings in which the iteration is implemented as a least fixpoint. We implement them based on Kozen's axioms [13], but most results are inherited from Conway semirings and iterings.

* Kleene Relation Algebras introduces Stone-Kleene relation algebras, which combine Stone relation algebras and Kleene algebras. This is similar to relation algebras with transitive closure [16] but allows us to talk about reachability in weighted graphs. Many results in this theory are useful for verifying the correctness of Prim's and Kruskal's minimum spanning tree algorithms.

* Subalgebras of Kleene Relation Algebras studies the regular elements of a Stone-Kleene relation algebra and shows that they form a Kleene relation subalgebra.

* Matrix Kleene Algebras lifts the Kleene star to finite square matrices using Conway's automata-based construction. This involves an operation to restrict matrices to specific indices and a calculus for such restrictions. An implementation for the Kleene star of matrices was given in [3] without proof; this is the first formally verified correctness proof.

The development is based on a theory of Stone relation algebras [11, 12]. We apply Stone-Kleene relation algebras to verify Prim's minimum spanning tree algorithm in Isabelle/HOL in [10].

Related libraries for Kleene algebras, regular algebras and relation algebras in the Archive of Formal Proofs are [1, 2, 8]. Kleene algebras are covered in the theory `Kleene_Algebra/Kleene_Algebra.thy`, but unlike the present development it is not based on general algebras using simulation axioms, which are useful to describe various computation models. The theory `Regular_Algebras/Regular_Algebras.thy` compares different axiomatisations of regular algebras. The theory `Kleene_Algebra/Matrix.thy` covers matrices over dioids, but does not implement the Kleene star of matrices. The theory `Relation_Algebra/Relation_Algebra_RTC.thy` combines Kleene algebras and relation algebras, but is very limited in scope and not applicable as we need the weaker axioms of Stone relation algebras.

2 Iterings

This theory introduces algebraic structures with an operation that describes iteration in various relational computation models. An iteration describes the repeated sequential execution of a computation. This is typically modelled by fixpoints, but different computation models use different fixpoints in the refinement order. We therefore look at equational and simulation axioms rather than induction axioms. Our development is based on [9] and the proposed algebras generalise Kleene algebras.

We first consider a variant of Conway semirings [5] based on idempotent left semirings. Conway semirings expand semirings by an iteration operation satisfying Conway's sumstar and productstar axioms [7]. Many properties of iteration follow already from these equational axioms.

Next we introduce iterings, which use generalised versions of simulation axioms in addition to sumstar and productstar. Unlike the induction axioms of the Kleene star, which hold only in partial-correctness models, the simulation axioms are also valid in total and general correctness models. They are still powerful enough to prove the correctness of complex results such as separation theorems of [6] and Back's atomicity refinement theorem [4, 17].

theory *Iterings*

imports *Stone-Relation-Algebras.Semirings*

begin

2.1 Conway Semirings

In this section, we consider equational axioms for iteration. The algebraic structures are based on idempotent left semirings, which are expanded by a unary iteration operation. We start with an unfold property, one inequality of the sliding rule and distributivity over joins, which is similar to Conway's sumstar.

class *circ* =
  fixes *circ* :: $'a \Rightarrow 'a$ (‹-°› [*100*] *100*)

class *left-conway-semiring* = *idempotent-left-semiring* + *circ* +
assumes *circ-left-unfold*: $1 \sqcup x * x^\circ = x^\circ$
assumes *circ-left-slide*: $(x * y)^\circ * x \leq x * (y * x)^\circ$
assumes *circ-sup-1*: $(x \sqcup y)^\circ = x^\circ * (y * x^\circ)^\circ$
begin

We obtain one inequality of Conway's productstar, as well as of the other unfold rule.

lemma *circ-mult-sub*:
$1 \sqcup x * (y * x)^\circ * y \leq (x * y)^\circ$
by (*metis sup-right-isotone circ-left-slide circ-left-unfold mult-assoc mult-right-isotone*)

lemma *circ-right-unfold-sub*:
$1 \sqcup x^\circ * x \leq x^\circ$
by (*metis circ-mult-sub mult-1-left mult-1-right*)

lemma *circ-zero*:
$bot^\circ = 1$
by (*metis sup-monoid.add-0-right circ-left-unfold mult-left-zero*)

lemma *circ-increasing*:
$x \leq x^\circ$
by (*metis le-supI2 circ-left-unfold circ-right-unfold-sub mult-1-left mult-right-sub-dist-sup-left order-trans*)

lemma *circ-reflexive*:
$1 \leq x^\circ$
by (*metis sup-left-divisibility circ-left-unfold*)

lemma *circ-mult-increasing*:
$x \leq x * x^\circ$
by (*metis circ-reflexive mult-right-isotone mult-1-right*)

lemma *circ-mult-increasing-2*:
$x \leq x^\circ * x$
by (*metis circ-reflexive mult-left-isotone mult-1-left*)

lemma *circ-transitive-equal*:
$x^\circ * x^\circ = x^\circ$
by (*metis sup-idem circ-sup-1 circ-left-unfold mult-assoc*)

While iteration is not idempotent, a fixpoint is reached after applying this operation twice. Iteration is idempotent for the unit.

lemma *circ-circ-circ*:
$x^{\circ\circ\circ} = x^{\circ\circ}$
by (*metis sup-idem circ-sup-1 circ-increasing circ-transitive-equal le-iff-sup*)

lemma *circ-one*:

$1^\circ = 1^{\circ\circ}$
by (*metis circ-circ-circ circ-zero*)

lemma *circ-sup-sub*:
$(x^\circ * y)^\circ * x^\circ \le (x \sqcup y)^\circ$
by (*metis circ-sup-1 circ-left-slide*)

lemma *circ-plus-one*:
$x^\circ = 1 \sqcup x^\circ$
by (*metis le-iff-sup circ-reflexive*)

Iteration satisfies a characteristic property of reflexive transitive closures.

lemma *circ-rtc-2*:
$1 \sqcup x \sqcup x^\circ * x^\circ = x^\circ$
by (*metis sup-assoc circ-increasing circ-plus-one circ-transitive-equal le-iff-sup*)

lemma *mult-zero-circ*:
$(x * bot)^\circ = 1 \sqcup x * bot$
by (*metis circ-left-unfold mult-assoc mult-left-zero*)

lemma *mult-zero-sup-circ*:
$(x \sqcup y * bot)^\circ = x^\circ * (y * bot)^\circ$
by (*metis circ-sup-1 mult-assoc mult-left-zero*)

lemma *circ-plus-sub*:
$x^\circ * x \le x * x^\circ$
by (*metis circ-left-slide mult-1-left mult-1-right*)

lemma *circ-loop-fixpoint*:
$y * (y^\circ * z) \sqcup z = y^\circ * z$
by (*metis sup-commute circ-left-unfold mult-assoc mult-1-left mult-right-dist-sup*)

lemma *left-plus-below-circ*:
$x * x^\circ \le x^\circ$
by (*metis sup.coboundedl2 circ-left-unfold*)

lemma *right-plus-below-circ*:
$x^\circ * x \le x^\circ$
using *circ-right-unfold-sub* **by** *auto*

lemma *circ-sup-upper-bound*:
$x \le z^\circ \Longrightarrow y \le z^\circ \Longrightarrow x \sqcup y \le z^\circ$
by *simp*

lemma *circ-mult-upper-bound*:
$x \le z^\circ \Longrightarrow y \le z^\circ \Longrightarrow x * y \le z^\circ$
by (*metis mult-isotone circ-transitive-equal*)

lemma *circ-sub-dist*:
$x^\circ \leq (x \sqcup y)^\circ$
by (*metis circ-sup-sub circ-plus-one mult-1-left mult-right-sub-dist-sup-left order-trans*)

lemma *circ-sub-dist-1*:
$x \leq (x \sqcup y)^\circ$
using *circ-increasing le-supE* **by** *blast*

lemma *circ-sub-dist-2*:
$x * y \leq (x \sqcup y)^\circ$
by (*metis sup-commute circ-mult-upper-bound circ-sub-dist-1*)

lemma *circ-sub-dist-3*:
$x^\circ * y^\circ \leq (x \sqcup y)^\circ$
by (*metis sup-commute circ-mult-upper-bound circ-sub-dist*)

lemma *circ-isotone*:
$x \leq y \Longrightarrow x^\circ \leq y^\circ$
by (*metis circ-sub-dist le-iff-sup*)

lemma *circ-sup-2*:
$(x \sqcup y)^\circ \leq (x^\circ * y^\circ)^\circ$
by (*metis sup.bounded-iff circ-increasing circ-isotone circ-reflexive mult-isotone mult-1-left mult-1-right*)

lemma *circ-sup-one-left-unfold*:
$1 \leq x \Longrightarrow x * x^\circ = x^\circ$
by (*metis order.antisym le-iff-sup mult-1-left mult-right-sub-dist-sup-left left-plus-below-circ*)

lemma *circ-sup-one-right-unfold*:
$1 \leq x \Longrightarrow x^\circ * x = x^\circ$
by (*metis order.antisym le-iff-sup mult-left-sub-dist-sup-left mult-1-right right-plus-below-circ*)

lemma *circ-decompose-4*:
$(x^\circ * y^\circ)^\circ = x^\circ * (y^\circ * x^\circ)^\circ$
by (*metis sup-assoc sup-commute circ-sup-1 circ-loop-fixpoint circ-plus-one circ-rtc-2 circ-transitive-equal mult-assoc*)

lemma *circ-decompose-5*:
$(x^\circ * y^\circ)^\circ = (y^\circ * x^\circ)^\circ$
by (*metis circ-decompose-4 circ-loop-fixpoint order.antisym mult-right-sub-dist-sup-right mult-assoc*)

lemma *circ-decompose-6*:
$x^\circ * (y * x^\circ)^\circ = y^\circ * (x * y^\circ)^\circ$
by (*metis sup-commute circ-sup-1*)

lemma *circ-decompose-7*:
$(x \sqcup y)^\circ = x^\circ * y^\circ * (x \sqcup y)^\circ$
by (*metis circ-sup-1 circ-decompose-6 circ-transitive-equal mult-assoc*)

lemma *circ-decompose-8*:
$(x \sqcup y)^\circ = (x \sqcup y)^\circ * x^\circ * y^\circ$
by (*metis order.antisym eq-refl mult-assoc mult-isotone mult-1-right circ-mult-upper-bound circ-reflexive circ-sub-dist-3*)

lemma *circ-decompose-9*:
$(x^\circ * y^\circ)^\circ = x^\circ * y^\circ * (x^\circ * y^\circ)^\circ$
by (*metis circ-decompose-4 mult-assoc*)

lemma *circ-decompose-10*:
$(x^\circ * y^\circ)^\circ = (x^\circ * y^\circ)^\circ * x^\circ * y^\circ$
by (*metis sup-ge2 circ-loop-fixpoint circ-reflexive circ-sup-one-right-unfold mult-assoc order-trans*)

lemma *circ-back-loop-prefixpoint*:
$(z * y^\circ) * y \sqcup z \leq z * y^\circ$
by (*metis sup.bounded-iff circ-left-unfold mult-assoc mult-left-sub-dist-sup-left mult-right-isotone mult-1-right right-plus-below-circ*)

We obtain the fixpoint and prefixpoint properties of iteration, but not least or greatest fixpoint properties.

lemma *circ-loop-is-fixpoint*:
is-fixpoint $(\lambda x\ .\ y * x \sqcup z)\ (y^\circ * z)$
by (*metis circ-loop-fixpoint is-fixpoint-def*)

lemma *circ-back-loop-is-prefixpoint*:
is-prefixpoint $(\lambda x\ .\ x * y \sqcup z)\ (z * y^\circ)$
by (*metis circ-back-loop-prefixpoint is-prefixpoint-def*)

lemma *circ-circ-sup*:
$(1 \sqcup x)^\circ = x^{\circ\circ}$
by (*metis sup-commute circ-sup-1 circ-decompose-4 circ-zero mult-1-right*)

lemma *circ-circ-mult-sub*:
$x^\circ * 1^\circ \leq x^{\circ\circ}$
by (*metis circ-increasing circ-isotone circ-mult-upper-bound circ-reflexive*)

lemma *left-plus-circ*:
$(x * x^\circ)^\circ = x^\circ$
by (*metis circ-left-unfold circ-sup-1 mult-1-right mult-sub-right-one sup.absorb1 mult-assoc*)

lemma *right-plus-circ*:
$(x^\circ * x)^\circ = x^\circ$

by (*metis sup-commute circ-isotone circ-loop-fixpoint circ-plus-sub circ-sub-dist order.eq-iff left-plus-circ*)

lemma *circ-square*:
$(x * x)^\circ \leq x^\circ$
by (*metis circ-increasing circ-isotone left-plus-circ mult-right-isotone*)

lemma *circ-mult-sub-sup*:
$(x * y)^\circ \leq (x \sqcup y)^\circ$
by (*metis sup-ge1 sup-ge2 circ-isotone circ-square mult-isotone order-trans*)

lemma *circ-sup-mult-zero*:
$x^\circ * y = (x \sqcup y * bot)^\circ * y$
proof −
have $(x \sqcup y * bot)^\circ * y = x^\circ * (1 \sqcup y * bot) * y$
by (*metis mult-zero-sup-circ mult-zero-circ*)
also have ... $= x^\circ * (y \sqcup y * bot)$
by (*metis mult-assoc mult-1-left mult-left-zero mult-right-dist-sup*)
also have ... $= x^\circ * y$
by (*metis sup-commute le-iff-sup zero-right-mult-decreasing*)
finally show *?thesis*
by *simp*
qed

lemma *troeger-1*:
$(x \sqcup y)^\circ = x^\circ * (1 \sqcup y * (x \sqcup y)^\circ)$
by (*metis circ-sup-1 circ-left-unfold mult-assoc*)

lemma *troeger-2*:
$(x \sqcup y)^\circ * z = x^\circ * (y * (x \sqcup y)^\circ * z \sqcup z)$
by (*metis circ-sup-1 circ-loop-fixpoint mult-assoc*)

lemma *troeger-3*:
$(x \sqcup y * bot)^\circ = x^\circ * (1 \sqcup y * bot)$
by (*metis mult-zero-sup-circ mult-zero-circ*)

lemma *circ-sup-sub-sup-one-1*:
$x \sqcup y \leq x^\circ * (1 \sqcup y)$
by (*metis circ-increasing circ-left-unfold mult-1-left mult-1-right mult-left-sub-dist-sup mult-right-sub-dist-sup-left order-trans sup-mono*)

lemma *circ-sup-sub-sup-one-2*:
$x^\circ * (x \sqcup y) \leq x^\circ * (1 \sqcup y)$
by (*metis circ-sup-sub-sup-one-1 circ-transitive-equal mult-assoc mult-right-isotone*)

lemma *circ-sup-sub-sup-one*:
$x * x^\circ * (x \sqcup y) \leq x * x^\circ * (1 \sqcup y)$
by (*metis circ-sup-sub-sup-one-2 mult-assoc mult-right-isotone*)

lemma *circ-square-2*:
$(x * x)^\circ * (x \sqcup 1) \leq x^\circ$
by (*metis sup.bounded-iff circ-increasing circ-mult-upper-bound circ-reflexive circ-square*)

lemma *circ-extra-circ*:
$(y * x^\circ)^\circ = (y * y^\circ * x^\circ)^\circ$
by (*metis circ-decompose-6 circ-transitive-equal left-plus-circ mult-assoc*)

lemma *circ-circ-sub-mult*:
$1^\circ * x^\circ \leq x^{\circ\circ}$
by (*metis circ-increasing circ-isotone circ-mult-upper-bound circ-reflexive*)

lemma *circ-decompose-11*:
$(x^\circ * y^\circ)^\circ = (x^\circ * y^\circ)^\circ * x^\circ$
by (*metis circ-decompose-10 circ-decompose-4 circ-decompose-5 circ-decompose-9 left-plus-circ*)

lemma *circ-mult-below-circ-circ*:
$(x * y)^\circ \leq (x^\circ * y)^\circ * x^\circ$
by (*metis circ-increasing circ-isotone circ-reflexive dual-order.trans mult-left-isotone mult-right-isotone mult-1-right*)

lemma *power-below-circ*:
power x $i \leq x^\circ$
apply (*induct rule*: *nat.induct*)
apply (*simp add*: *circ-reflexive*)
by (*simp add*: *circ-increasing circ-mult-upper-bound*)

end

The next class considers the interaction of iteration with a greatest element.

class *bounded-left-conway-semiring* = *bounded-idempotent-left-semiring* + *left-conway-semiring*
begin

lemma *circ-top*:
$top^\circ = top$
by (*simp add*: *order.antisym circ-increasing*)

lemma *circ-right-top*:
$x^\circ * top = top$
by (*metis sup-right-top circ-loop-fixpoint*)

lemma *circ-left-top*:

$top * x^\circ = top$
by (*metis circ-right-top circ-top circ-decompose-11*)

lemma *mult-top-circ*:
$(x * top)^\circ = 1 \sqcup x * top$
by (*metis circ-left-top circ-left-unfold mult-assoc*)

end

class *left-zero-conway-semiring* = *idempotent-left-zero-semiring* + *left-conway-semiring*
begin

lemma *mult-zero-sup-circ-2*:
$(x \sqcup y * bot)^\circ = x^\circ \sqcup x^\circ * y * bot$
by (*metis mult-assoc mult-left-dist-sup mult-1-right troeger-3*)

lemma *circ-unfold-sum*:
$(x \sqcup y)^\circ = x^\circ \sqcup x^\circ * y * (x \sqcup y)^\circ$
by (*metis mult-assoc mult-left-dist-sup mult-1-right troeger-1*)

end

The next class assumes the full sliding equation.

class *left-conway-semiring-1* = *left-conway-semiring* +
assumes *circ-right-slide*: $x * (y * x)^\circ \leq (x * y)^\circ * x$
begin

lemma *circ-slide-1*:
$x * (y * x)^\circ = (x * y)^\circ * x$
by (*metis order.antisym circ-left-slide circ-right-slide*)

This implies the full unfold rules and Conway's productstar.

lemma *circ-right-unfold-1*:
$1 \sqcup x^\circ * x = x^\circ$
by (*metis circ-left-unfold circ-slide-1 mult-1-left mult-1-right*)

lemma *circ-mult-1*:
$(x * y)^\circ = 1 \sqcup x * (y * x)^\circ * y$
by (*metis circ-left-unfold circ-slide-1 mult-assoc*)

lemma *circ-sup-9*:
$(x \sqcup y)^\circ = (x^\circ * y)^\circ * x^\circ$
by (*metis circ-sup-1 circ-slide-1*)

lemma *circ-plus-same*:
$x^\circ * x = x * x^\circ$
by (*metis circ-slide-1 mult-1-left mult-1-right*)

lemma *circ-decompose-12*:
$x^\circ * y^\circ \leq (x^\circ * y)^\circ * x^\circ$
by (*metis circ-sup-9 circ-sub-dist-3*)

end

class *left-zero-conway-semiring-1* = *left-zero-conway-semiring* + *left-conway-semiring-1*
begin

lemma *circ-back-loop-fixpoint*:
$(z * y^\circ) * y \sqcup z = z * y^\circ$
by (*metis sup-commute circ-left-unfold circ-plus-same mult-assoc mult-left-dist-sup mult-1-right*)

lemma *circ-back-loop-is-fixpoint*:
is-fixpoint $(\lambda x\ .\ x * y \sqcup z)\ (z * y^\circ)$
by (*metis circ-back-loop-fixpoint is-fixpoint-def*)

lemma *circ-elimination*:
$x * y = bot \Longrightarrow x * y^\circ \leq x$
by (*metis sup-monoid.add-0-left circ-back-loop-fixpoint circ-plus-same mult-assoc mult-left-zero order-refl*)

end

2.2 Iterings

This section adds simulation axioms to Conway semirings. We consider several classes with increasingly general simulation axioms.

class *itering-1* = *left-conway-semiring-1* +
assumes *circ-simulate*: $z * x \leq y * z \longrightarrow z * x^\circ \leq y^\circ * z$
begin

lemma *circ-circ-mult*:
$1^\circ * x^\circ = x^{\circ\circ}$
by (*metis order.antisym circ-circ-sup circ-reflexive circ-simulate circ-sub-dist-3 circ-sup-one-left-unfold circ-transitive-equal mult-1-left order-refl*)

lemma *sub-mult-one-circ*:
$x * 1^\circ \leq 1^\circ * x$
by (*metis circ-simulate mult-1-left mult-1-right order-refl*)

The left simulation axioms is enough to prove a basic import property of tests.

lemma *circ-import*:
assumes $p \leq p * p$
and $p \leq 1$
and $p * x \leq x * p$

shows $p * x^\circ = p * (p * x)^\circ$
proof –
have $p * x \leq p * (p * x * p) * p$
by (*metis assms coreflexive-transitive order.eq-iff test-preserves-equation mult-assoc*)
hence $p * x^\circ \leq p * (p * x)^\circ$
by (*metis* (*no-types*) *assms circ-simulate circ-slide-1 test-preserves-equation*)
thus *?thesis*
by (*metis assms*(*2*) *circ-isotone mult-left-isotone mult-1-left mult-right-isotone order.antisym*)
qed

end

Including generalisations of both simulation axioms allows us to prove separation rules.

class *itering-2* = *left-conway-semiring-1* +
assumes *circ-simulate-right*: $z * x \leq y * z \sqcup w \longrightarrow z * x^\circ \leq y^\circ * (z \sqcup w * x^\circ)$
assumes *circ-simulate-left*: $x * z \leq z * y \sqcup w \longrightarrow x^\circ * z \leq (z \sqcup x^\circ * w) * y^\circ$
begin

subclass *itering-1*
apply *unfold-locales*
by (*metis sup-monoid.add-0-right circ-simulate-right mult-left-zero*)

lemma *circ-simulate-left-1*:
$x * z \leq z * y \Longrightarrow x^\circ * z \leq z * y^\circ \sqcup x^\circ * bot$
by (*metis sup-monoid.add-0-right circ-simulate-left mult-assoc mult-left-zero mult-right-dist-sup*)

lemma *circ-separate-1*:
assumes $y * x \leq x * y$
shows $(x \sqcup y)^\circ = x^\circ * y^\circ$
proof –
have $y^\circ * x \leq x * y^\circ \sqcup y^\circ * bot$
by (*metis assms circ-simulate-left-1*)
hence $y^\circ * x * y^\circ \leq x * y^\circ * y^\circ \sqcup y^\circ * bot * y^\circ$
by (*metis mult-assoc mult-left-isotone mult-right-dist-sup*)
also have ... $= x * y^\circ \sqcup y^\circ * bot$
by (*metis circ-transitive-equal mult-assoc mult-left-zero*)
finally have $y^\circ * (x * y^\circ)^\circ \leq x^\circ * (y^\circ \sqcup y^\circ * bot)$
using *circ-simulate-right mult-assoc* **by** *fastforce*
also have ... $= x^\circ * y^\circ$
by (*simp add*: *sup-absorb1 zero-right-mult-decreasing*)
finally have $(x \sqcup y)^\circ \leq x^\circ * y^\circ$
by (*simp add*: *circ-decompose-6 circ-sup-1*)
thus *?thesis*
by (*simp add*: *order.antisym circ-sub-dist-3*)
qed

lemma *circ-circ-mult-1*:
$x^\circ * 1^\circ = x^{\circ\circ}$
by (*metis sup-commute circ-circ-sup circ-separate-1 mult-1-left mult-1-right order-refl*)

end

With distributivity, we also get Back's atomicity refinement theorem.

class *itering-3* = *itering-2* + *left-zero-conway-semiring-1*
begin

lemma *circ-simulate-1*:
assumes $y * x \leq x * y$
shows $y^\circ * x^\circ \leq x^\circ * y^\circ$
proof –
have $y * x^\circ \leq x^\circ * y$
by (*metis assms circ-simulate*)
hence $y^\circ * x^\circ \leq x^\circ * y^\circ \sqcup y^\circ * bot$
by (*metis circ-simulate-left-1*)
thus *?thesis*
by (*metis sup-assoc sup-monoid.add-0-right circ-loop-fixpoint mult-assoc mult-left-zero mult-zero-sup-circ-2*)
qed

lemma *atomicity-refinement*:
assumes $s = s * q$
and $x = q * x$
and $q * b = bot$
and $r * b \leq b * r$
and $r * l \leq l * r$
and $x * l \leq l * x$
and $b * l \leq l * b$
and $q * l \leq l * q$
and $r^\circ * q \leq q * r^\circ$
and $q \leq 1$
shows $s * (x \sqcup b \sqcup r \sqcup l)^\circ * q \leq s * (x * b^\circ * q \sqcup r \sqcup l)^\circ$
proof –
have $(x \sqcup b \sqcup r) * l \leq l * (x \sqcup b \sqcup r)$
using *assms*$(5-7)$ *mult-left-dist-sup mult-right-dist-sup semiring.add-mono*
by *presburger*
hence $s * (x \sqcup b \sqcup r \sqcup l)^\circ * q = s * l^\circ * (x \sqcup b \sqcup r)^\circ * q$
by (*metis sup-commute circ-separate-1 mult-assoc*)
also have ... $= s * l^\circ * b^\circ * r^\circ * q * (x * b^\circ * r^\circ * q)^\circ$
proof –
have $(b \sqcup r)^\circ = b^\circ * r^\circ$
by (*simp add*: *assms*(*4*) *circ-separate-1*)
hence $b^\circ * r^\circ * (q * (x * b^\circ * r^\circ))^\circ = (x \sqcup b \sqcup r)^\circ$
by (*metis* (*full-types*) *assms*(*2*) *circ-sup-1 sup-assoc sup-commute mult-assoc*)

thus *?thesis*
by (*metis circ-slide-1 mult-assoc*)
qed
also have ... $\leq s * l^\circ * b^\circ * r^\circ * q * (x * b^\circ * q * r^\circ)^\circ$
by (*metis assms*(*9*) *circ-isotone mult-assoc mult-right-isotone*)
also have ... $\leq s * q * l^\circ * b^\circ * r^\circ * (x * b^\circ * q * r^\circ)^\circ$
by (*metis assms*(*1*,*10*) *mult-left-isotone mult-right-isotone mult-1-right*)
also have ... $\leq s * l^\circ * q * b^\circ * r^\circ * (x * b^\circ * q * r^\circ)^\circ$
by (*metis assms*(*1*,*8*) *circ-simulate mult-assoc mult-left-isotone mult-right-isotone*)
also have ... $\leq s * l^\circ * r^\circ * (x * b^\circ * q * r^\circ)^\circ$
by (*metis assms*(*3*,*10*) *sup-monoid.add-0-left circ-back-loop-fixpoint circ-plus-same mult-assoc mult-left-zero mult-left-isotone mult-right-isotone mult-1-right*)
also have ... $\leq s * (x * b^\circ * q \sqcup r \sqcup l)^\circ$
by (*metis sup-commute circ-sup-1 circ-sub-dist-3 mult-assoc mult-right-isotone*)
finally show *?thesis*
.
qed

end

The following class contains the most general simulation axioms we consider. They allow us to prove further separation properties.

class *itering* = *idempotent-left-zero-semiring* + *circ* +
assumes *circ-sup*: $(x \sqcup y)^\circ = (x^\circ * y)^\circ * x^\circ$
assumes *circ-mult*: $(x * y)^\circ = 1 \sqcup x * (y * x)^\circ * y$
assumes *circ-simulate-right-plus*: $z * x \leq y * y^\circ * z \sqcup w \longrightarrow z * x^\circ \leq y^\circ * (z \sqcup w * x^\circ)$
assumes *circ-simulate-left-plus*: $x * z \leq z * y^\circ \sqcup w \longrightarrow x^\circ * z \leq (z \sqcup x^\circ * w) * y^\circ$
begin

lemma *circ-right-unfold*:
$1 \sqcup x^\circ * x = x^\circ$
by (*metis circ-mult mult-1-left mult-1-right*)

lemma *circ-slide*:
$x * (y * x)^\circ = (x * y)^\circ * x$
proof −
have $x * (y * x)^\circ = Rf\ x\ (y * 1 \sqcup y * (x * (y * x)^\circ * y)) * x$
by (*metis* (*no-types*) *circ-mult mult-1-left mult-1-right mult-left-dist-sup mult-right-dist-sup mult-assoc*)
thus *?thesis*
by (*metis* (*no-types*) *circ-mult mult-1-right mult-left-dist-sup mult-assoc*)
qed

subclass *itering-3*

apply *unfold-locales*
apply (*metis circ-mult mult-1-left mult-1-right*)
apply (*metis circ-slide order-refl*)
apply (*metis circ-sup circ-slide*)
apply (*metis circ-slide order-refl*)
apply (*metis sup-left-isotone circ-right-unfold mult-left-isotone mult-left-sub-dist-sup-left mult-1-right order-trans circ-simulate-right-plus*)
by (*metis sup-commute sup-ge1 sup-right-isotone circ-mult mult-right-isotone mult-1-right order-trans circ-simulate-left-plus*)

lemma *circ-simulate-right-plus-1*:
$z * x \le y * y^\circ * z \Longrightarrow z * x^\circ \le y^\circ * z$
by (*metis sup-monoid.add-0-right circ-simulate-right-plus mult-left-zero*)

lemma *circ-simulate-left-plus-1*:
$x * z \le z * y^\circ \Longrightarrow x^\circ * z \le z * y^\circ \sqcup x^\circ * bot$
by (*metis sup-monoid.add-0-right circ-simulate-left-plus mult-assoc mult-left-zero mult-right-dist-sup*)

lemma *circ-simulate-2*:
$y * x^\circ \le x^\circ * y^\circ \longleftrightarrow y^\circ * x^\circ \le x^\circ * y^\circ$
apply (*rule iffI*)
apply (*metis sup-assoc sup-monoid.add-0-right circ-loop-fixpoint circ-simulate-left-plus-1 mult-assoc mult-left-zero mult-zero-sup-circ-2*)
by (*metis circ-increasing mult-left-isotone order-trans*)

lemma *circ-simulate-absorb*:
$y * x \le x \Longrightarrow y^\circ * x \le x \sqcup y^\circ * bot$
by (*metis circ-simulate-left-plus-1 circ-zero mult-1-right*)

lemma *circ-simulate-3*:
$y * x^\circ \le x^\circ \Longrightarrow y^\circ * x^\circ \le x^\circ * y^\circ$
by (*metis sup.bounded-iff circ-reflexive circ-simulate-2 le-iff-sup mult-right-isotone mult-1-right*)

lemma *circ-separate-mult-1*:
$y * x \le x * y \Longrightarrow (x * y)^\circ \le x^\circ * y^\circ$
by (*metis circ-mult-sub-sup circ-separate-1*)

lemma *circ-separate-unfold*:
$(y * x^\circ)^\circ = y^\circ \sqcup y^\circ * y * x * x^\circ * (y * x^\circ)^\circ$
by (*metis circ-back-loop-fixpoint circ-plus-same circ-unfold-sum sup-commute mult-assoc*)

lemma *separation*:
assumes $y * x \le x * y^\circ$
shows $(x \sqcup y)^\circ = x^\circ * y^\circ$
proof −
have $y^\circ * x * y^\circ \le x * y^\circ \sqcup y^\circ * bot$

by (*metis assms circ-simulate-left-plus-1 circ-transitive-equal mult-assoc mult-left-isotone*)
thus *?thesis*
by (*metis sup-commute circ-sup-1 circ-simulate-right circ-sub-dist-3 le-iff-sup mult-assoc mult-left-zero zero-right-mult-decreasing*)
qed

lemma *simulation*:
$y * x \leq x * y^\circ \Longrightarrow y^\circ * x^\circ \leq x^\circ * y^\circ$
by (*metis sup-ge2 circ-isotone circ-mult-upper-bound circ-sub-dist separation*)

lemma *circ-simulate-4*:
assumes $y * x \leq x * x^\circ * (1 \sqcup y)$
shows $y^\circ * x^\circ \leq x^\circ * y^\circ$
proof −
have $x \sqcup (x * x^\circ * x * x \sqcup x * x) = x * x^\circ$
by (*metis* (*no-types*) *circ-back-loop-fixpoint mult-right-dist-sup sup-commute*)
hence $x \leq x * x^\circ * 1 \sqcup x * x^\circ * y$
by (*metis mult-1-right sup-assoc sup-ge1*)
hence $(1 \sqcup y) * x \leq x * x^\circ * (1 \sqcup y)$
using *assms mult-left-dist-sup mult-right-dist-sup* **by** *force*
hence $y * x^\circ \leq x^\circ * y^\circ$
by (*metis circ-sup-upper-bound circ-increasing circ-reflexive circ-simulate-right-plus-1 mult-right-isotone mult-right-sub-dist-sup-right order-trans*)
thus *?thesis*
by (*metis circ-simulate-2*)
qed

lemma *circ-simulate-5*:
$y * x \leq x * x^\circ * (x \sqcup y) \Longrightarrow y^\circ * x^\circ \leq x^\circ * y^\circ$
by (*metis circ-sup-sub-sup-one circ-simulate-4 order-trans*)

lemma *circ-simulate-6*:
$y * x \leq x * (x \sqcup y) \Longrightarrow y^\circ * x^\circ \leq x^\circ * y^\circ$
by (*metis sup-commute circ-back-loop-fixpoint circ-simulate-5 mult-right-sub-dist-sup-left order-trans*)

lemma *circ-separate-4*:
assumes $y * x \leq x * x^\circ * (1 \sqcup y)$
shows $(x \sqcup y)^\circ = x^\circ * y^\circ$
proof −
have $y * x * x^\circ \leq x * x^\circ * (1 \sqcup y) * x^\circ$
by (*simp add*: *assms mult-left-isotone*)
also have ... $= x * x^\circ \sqcup x * x^\circ * y * x^\circ$
by (*simp add*: *circ-transitive-equal mult-left-dist-sup mult-right-dist-sup mult-assoc*)
also have ... $\leq x * x^\circ \sqcup x * x^\circ * x^\circ * y^\circ$
by (*metis assms sup-right-isotone circ-simulate-2 circ-simulate-4 mult-assoc*

mult-right-isotone)
finally have $y * x * x^\circ \leq x * x^\circ * y^\circ$
by (*metis circ-reflexive circ-transitive-equal le-iff-sup mult-assoc mult-right-isotone mult-1-right*)
thus *?thesis*
by (*metis circ-sup-1 left-plus-circ mult-assoc separation*)
qed

lemma *circ-separate-5*:
$y * x \leq x * x^\circ * (x \sqcup y) \Longrightarrow (x \sqcup y)^\circ = x^\circ * y^\circ$
by (*metis circ-sup-sub-sup-one circ-separate-4 order-trans*)

lemma *circ-separate-6*:
$y * x \leq x * (x \sqcup y) \Longrightarrow (x \sqcup y)^\circ = x^\circ * y^\circ$
by (*metis sup-commute circ-back-loop-fixpoint circ-separate-5 mult-right-sub-dist-sup-left order-trans*)

end

class *bounded-itering* = *bounded-idempotent-left-zero-semiring* + *itering*
begin

subclass *bounded-left-conway-semiring* **..**

end

We finally expand Conway semirings and iterings by an element that corresponds to the endless loop.

class *L* =
fixes *L* :: *'a*

class *left-conway-semiring-L* = *left-conway-semiring* + *L* +
assumes *one-circ-mult-split*: $1^\circ * x = L \sqcup x$
assumes *L-split-sup*: $x * (y \sqcup L) \leq x * y \sqcup L$
begin

lemma *L-def*:
$L = 1^\circ * bot$
by (*metis sup-monoid.add-0-right one-circ-mult-split*)

lemma *one-circ-split*:
$1^\circ = L \sqcup 1$
by (*metis mult-1-right one-circ-mult-split*)

lemma *one-circ-circ-split*:
$1^{\circ\circ} = L \sqcup 1$
by (*metis circ-one one-circ-split*)

lemma *sub-mult-one-circ*:
$x * 1^\circ \leq 1^\circ * x$
by (*metis L-split-sup sup-commute mult-1-right one-circ-mult-split*)

lemma *one-circ-mult-split-2*:
$1^\circ * x = x * 1^\circ \sqcup L$
proof −
have *1*: $x * 1^\circ \leq L \sqcup x$
using *one-circ-mult-split sub-mult-one-circ* **by** *presburger*
have $x \sqcup x * 1^\circ = x * 1^\circ$
by (*meson circ-back-loop-prefixpoint le-iff-sup sup.boundedE*)
thus *?thesis*
using *1* **by** (*simp add*: *le-iff-sup one-circ-mult-split sup-assoc sup-commute*)
qed

lemma *sub-mult-one-circ-split*:
$x * 1^\circ \leq x \sqcup L$
by (*metis sup-commute one-circ-mult-split sub-mult-one-circ*)

lemma *sub-mult-one-circ-split-2*:
$x * 1^\circ \leq x \sqcup 1^\circ$
by (*metis L-def sup-right-isotone order-trans sub-mult-one-circ-split zero-right-mult-decreasing*)

lemma *L-split*:
$x * L \leq x * bot \sqcup L$
by (*metis L-split-sup sup-monoid.add-0-left*)

lemma *L-left-zero*:
$L * x = L$
by (*metis L-def mult-assoc mult-left-zero*)

lemma *one-circ-L*:
$1^\circ * L = L$
by (*metis L-def circ-transitive-equal mult-assoc*)

lemma *mult-L-circ*:
$(x * L)^\circ = 1 \sqcup x * L$
by (*metis L-left-zero circ-left-unfold mult-assoc*)

lemma *mult-L-circ-mult*:
$(x * L)^\circ * y = y \sqcup x * L$
by (*metis L-left-zero mult-L-circ mult-assoc mult-1-left mult-right-dist-sup*)

lemma *circ-L*:
$L^\circ = L \sqcup 1$
by (*metis L-left-zero sup-commute circ-left-unfold*)

lemma *L-below-one-circ*:
  $L \leq 1^\circ$
  by (*metis L-def zero-right-mult-decreasing*)

lemma *circ-circ-mult-1*:
  $x^\circ * 1^\circ = x^{\circ\circ}$
  by (*metis L-left-zero sup-commute circ-sup-1 circ-circ-sup mult-zero-circ one-circ-split*)

lemma *circ-circ-mult*:
  $1^\circ * x^\circ = x^{\circ\circ}$
  by (*metis order.antisym circ-circ-mult-1 circ-circ-sub-mult sub-mult-one-circ*)

lemma *circ-circ-split*:
  $x^{\circ\circ} = L \sqcup x^\circ$
  by (*metis circ-circ-mult one-circ-mult-split*)

lemma *circ-sup-6*:
  $L \sqcup (x \sqcup y)^\circ = (x^\circ * y^\circ)^\circ$
  by (*metis sup-assoc sup-commute circ-sup-1 circ-circ-sup circ-circ-split circ-decompose-4*)

end

class *itering-L* = *itering* + *L* +
  assumes *L-def*: $L = 1^\circ * bot$
begin

lemma *one-circ-split*:
  $1^\circ = L \sqcup 1$
  by (*metis L-def sup-commute order.antisym circ-sup-upper-bound circ-reflexive circ-simulate-absorb mult-1-right order-refl zero-right-mult-decreasing*)

lemma *one-circ-mult-split*:
  $1^\circ * x = L \sqcup x$
  by (*metis L-def sup-commute circ-loop-fixpoint mult-assoc mult-left-zero mult-zero-circ one-circ-split*)

lemma *sub-mult-one-circ-split*:
  $x * 1^\circ \leq x \sqcup L$
  by (*metis sup-commute one-circ-mult-split sub-mult-one-circ*)

lemma *sub-mult-one-circ-split-2*:
  $x * 1^\circ \leq x \sqcup 1^\circ$
  by (*metis L-def sup-right-isotone order-trans sub-mult-one-circ-split zero-right-mult-decreasing*)

lemma *L-split*:
  $x * L \leq x * bot \sqcup L$

by (*metis L-def mult-assoc mult-left-isotone mult-right-dist-sup sub-mult-one-circ-split-2*)

subclass *left-conway-semiring-L*
apply *unfold-locales*
apply (*metis L-def sup-commute circ-loop-fixpoint mult-assoc mult-left-zero mult-zero-circ one-circ-split*)
by (*metis sup-commute mult-assoc mult-left-isotone one-circ-mult-split sub-mult-one-circ*)

lemma *circ-left-induct-mult-L*:
$L \le x \Longrightarrow x * y \le x \Longrightarrow x * y^{\circ} \le x$
by (*metis circ-one circ-simulate le-iff-sup one-circ-mult-split*)

lemma *circ-left-induct-mult-iff-L*:
$L \le x \Longrightarrow x * y \le x \longleftrightarrow x * y^{\circ} \le x$
by (*metis sup.bounded-iff circ-back-loop-fixpoint circ-left-induct-mult-L le-iff-sup*)

lemma *circ-left-induct-L*:
$L \le x \Longrightarrow x * y \sqcup z \le x \Longrightarrow z * y^{\circ} \le x$
by (*metis sup.bounded-iff circ-left-induct-mult-L le-iff-sup mult-right-dist-sup*)

end

end

3 Kleene Algebras

Kleene algebras have been axiomatised by Kozen to describe the equational theory of regular languages [13]. Binary relations are another important model. This theory implements variants of Kleene algebras based on idempotent left semirings [15]. The weakening of some semiring axioms allows the treatment of further computation models. The presented algebras are special cases of iterings, so many results can be inherited.

theory *Kleene-Algebras*

imports *Iterings*

begin

We start with left Kleene algebras, which use the left unfold and left induction axioms of Kleene algebras.

class *star* =
fixes *star* :: $'a \Rightarrow 'a$ (‹-⋆› [*100*] *100*)

class *left-kleene-algebra* = *idempotent-left-semiring* + *star* +
assumes *star-left-unfold* : $1 \sqcup y * y^{\star} \le y^{\star}$

assumes *star-left-induct* : $z \sqcup y * x \leq x \longrightarrow y^\star * z \leq x$
begin

unbundle *no trancl-syntax*

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

lemma *star-left-unfold-equal*:
$1 \sqcup x * x^\star = x^\star$
by (*metis sup-right-isotone order.antisym mult-right-isotone mult-1-right star-left-induct star-left-unfold*)

This means that for some properties of Kleene algebras, only one inequality can be derived, as exemplified by the following sliding rule.

lemma *star-left-slide*:
$(x * y)^\star * x \leq x * (y * x)^\star$
by (*metis mult-assoc mult-left-sub-dist-sup mult-1-right star-left-induct star-left-unfold-equal*)

lemma *star-isotone*:
$x \leq y \Longrightarrow x^\star \leq y^\star$
by (*metis sup-right-isotone mult-left-isotone order-trans star-left-unfold mult-1-right star-left-induct*)

lemma *star-sup-1*:
$(x \sqcup y)^\star = x^\star * (y * x^\star)^\star$
proof (*rule order.antisym*)
have $y * x^\star * (y * x^\star)^\star \leq (y * x^\star)^\star$
using *sup-right-divisibility star-left-unfold-equal* **by** *auto*
also have ... $\leq x^\star * (y * x^\star)^\star$
using *mult-left-isotone sup-left-divisibility star-left-unfold-equal* **by** *fastforce*
finally have $(x \sqcup y) * (x^\star * (y * x^\star)^\star) \leq x^\star * (y * x^\star)^\star$
by (*metis le-supI mult-right-dist-sup mult-right-sub-dist-sup-right mult-assoc star-left-unfold-equal*)
hence $1 \sqcup (x \sqcup y) * (x^\star * (y * x^\star)^\star) \leq x^\star * (y * x^\star)^\star$
using *reflexive-mult-closed star-left-unfold* **by** *auto*
thus $(x \sqcup y)^\star \leq x^\star * (y * x^\star)^\star$
using *star-left-induct* **by** *force*
next
have $x^\star * (y * x^\star)^\star \leq x^\star * (y * (x \sqcup y)^\star)^\star$
by (*simp add*: *mult-right-isotone star-isotone*)
also have ... $\leq x^\star * ((x \sqcup y) * (x \sqcup y)^\star)^\star$
by (*simp add*: *mult-right-isotone mult-right-sub-dist-sup-right star-isotone*)
also have ... $\leq x^\star * (x \sqcup y)^{\star\star}$
using *mult-right-isotone star-left-unfold star-isotone* **by** *auto*
also have ... $\leq (x \sqcup y)^\star * (x \sqcup y)^{\star\star}$
by (*simp add*: *mult-left-isotone star-isotone*)
also have ... $\leq (x \sqcup y)^\star$
by (*metis sup.bounded-iff mult-1-right star-left-induct star-left-unfold*)

finally show $x^\star * (y * x^\star)^\star \le (x \sqcup y)^\star$
by *simp*
qed

lemma *plus-transitive*:
$x^+ * x^+ \le x^+$
by (*metis mult-right-isotone star-left-induct sup-absorb2 sup-ge2 mult-assoc star-left-unfold-equal*)

end

We now show that left Kleene algebras form iterings. A sublocale is used instead of a subclass, because iterings use a different iteration operation.

sublocale *left-kleene-algebra* < *star*: *left-conway-semiring* **where** *circ* = *star*
apply *unfold-locales*
apply (*rule star-left-unfold-equal*)
apply (*rule star-left-slide*)
by (*rule star-sup-1*)

context *left-kleene-algebra*
begin

A number of lemmas in this class are taken from Georg Struth's Kleene algebra theory [2].

lemma *star-sub-one*:
$x \le 1 \Longrightarrow x^\star = 1$
by (*metis sup-right-isotone order.eq-iff le-iff-sup mult-1-right star.circ-plus-one star-left-induct*)

lemma *star-one*:
$1^\star = 1$
by (*simp add*: *star-sub-one*)

lemma *star-left-induct-mult*:
$x * y \le y \Longrightarrow x^\star * y \le y$
by (*simp add*: *star-left-induct*)

lemma *star-left-induct-mult-iff*:
$x * y \le y \longleftrightarrow x^\star * y \le y$
using *mult-left-isotone order-trans star.circ-increasing star-left-induct-mult* **by** *blast*

lemma *star-involutive*:
$x^\star = x^{\star\star}$
using *star.circ-circ-sup star-sup-1 star-one* **by** *auto*

lemma *star-sup-one*:
$(1 \sqcup x)^\star = x^\star$
using *star.circ-circ-sup star-involutive* **by** *auto*

lemma *star-plus-loops*:
  $x^\star \sqcup 1 = x^+ \sqcup 1$
  using *star.circ-plus-one star-left-unfold-equal sup-commute* **by** *auto*

lemma *star-left-induct-equal*:
  $z \sqcup x * y = y \Longrightarrow x^\star * z \leq y$
  by (*simp add*: *star-left-induct*)

lemma *star-left-induct-mult-equal*:
  $x * y = y \Longrightarrow x^\star * y \leq y$
  by (*simp add*: *star-left-induct-mult*)

lemma *star-star-upper-bound*:
  $x^\star \leq z^\star \Longrightarrow x^{\star\star} \leq z^\star$
  using *star-involutive* **by** *auto*

lemma *star-simulation-left*:
  assumes $x * z \leq z * y$
    shows $x^\star * z \leq z * y^\star$
proof −
  have $x * z * y^\star \leq z * y * y^\star$
    by (*simp add*: *assms mult-left-isotone*)
  also have ... $\leq z * y^\star$
    by (*simp add*: *mult-right-isotone star.left-plus-below-circ mult-assoc*)
  finally have $z \sqcup x * z * y^\star \leq z * y^\star$
    using *star.circ-back-loop-prefixpoint* **by** *auto*
  thus *?thesis*
    by (*simp add*: *star-left-induct mult-assoc*)
qed

lemma *quasicomm-1*:
  $y * x \leq x * (x \sqcup y)^\star \longleftrightarrow y^\star * x \leq x * (x \sqcup y)^\star$
  by (*metis mult-isotone order-refl order-trans star.circ-increasing star-involutive star-simulation-left*)

lemma *star-rtc-3*:
  $1 \sqcup x \sqcup y * y = y \Longrightarrow x^\star \leq y$
  by (*metis sup.bounded-iff le-iff-sup mult-left-sub-dist-sup-left mult-1-right star-left-induct-mult-iff star.circ-sub-dist*)

lemma *star-decompose-1*:
  $(x \sqcup y)^\star = (x^\star * y^\star)^\star$
  apply (*rule order.antisym*)
  apply (*simp add*: *star.circ-sup-2*)
  using *star.circ-sub-dist-3 star-isotone star-involutive* **by** *fastforce*

lemma *star-sum*:
  $(x \sqcup y)^\star = (x^\star \sqcup y^\star)^\star$

using *star-decompose-1 star-involutive* **by** *auto*

lemma *star-decompose-3*:
$(x^\star * y^\star)^\star = x^\star * (y * x^\star)^\star$
using *star-sup-1 star-decompose-1* **by** *auto*

In contrast to iterings, we now obtain that the iteration operation results in least fixpoints.

lemma *star-loop-least-fixpoint*:
$y * x \sqcup z = x \Longrightarrow y^\star * z \le x$
by (*simp add*: *sup-commute star-left-induct-equal*)

lemma *star-loop-is-least-fixpoint*:
is-least-fixpoint $(\lambda x\ .\ y * x \sqcup z)\ (y^\star * z)$
by (*simp add*: *is-least-fixpoint-def star.circ-loop-fixpoint star-loop-least-fixpoint*)

lemma *star-loop-mu*:
$\mu\ (\lambda x\ .\ y * x \sqcup z) = y^\star * z$
by (*metis least-fixpoint-same star-loop-is-least-fixpoint*)

lemma *affine-has-least-fixpoint*:
has-least-fixpoint $(\lambda x\ .\ y * x \sqcup z)$
by (*metis has-least-fixpoint-def star-loop-is-least-fixpoint*)

lemma *star-outer-increasing*:
$x \le y^\star * x * y^\star$
by (*metis star.circ-back-loop-prefixpoint star.circ-loop-fixpoint sup.boundedE*)

end

We next add the right induction rule, which allows us to strengthen many inequalities of left Kleene algebras to equalities.

class *strong-left-kleene-algebra* = *left-kleene-algebra* +
assumes *star-right-induct*: $z \sqcup x * y \le x \longrightarrow z * y^\star \le x$
begin

lemma *star-plus*:
$y^\star * y = y * y^\star$
proof (*rule order.antisym*)
show $y^\star * y \le y * y^\star$
by (*simp add*: *star.circ-plus-sub*)
next
have $y^\star * y * y \le y^\star * y$
by (*simp add*: *mult-left-isotone star.right-plus-below-circ*)
hence $y \sqcup y^\star * y * y \le y^\star * y$
by (*simp add*: *star.circ-mult-increasing-2*)
thus $y * y^\star \le y^\star * y$

using *star-right-induct* **by** *blast*
qed

lemma *star-slide*:
$(x * y)^\star * x = x * (y * x)^\star$
proof (*rule order.antisym*)
show $(x * y)^\star * x \leq x * (y * x)^\star$
by (*rule star-left-slide*)
next
have $x \sqcup (x * y)^\star * x * y * x \leq (x * y)^\star * x$
by (*metis* (*full-types*) *sup.commute eq-refl star.circ-loop-fixpoint mult.assoc star-plus*)
thus $x * (y * x)^\star \leq (x * y)^\star * x$
by (*simp add*: *mult-assoc star-right-induct*)
qed

lemma *star-simulation-right*:
assumes $z * x \leq y * z$
shows $z * x^\star \leq y^\star * z$
proof −
have $y^\star * z * x \leq y^\star * z$
by (*metis assms dual-order.trans mult-isotone mult-left-sub-dist-sup-right star.circ-loop-fixpoint star.circ-transitive-equal sup.cobounded1 mult-assoc*)
thus *?thesis*
by (*metis le-supI star.circ-loop-fixpoint star-right-induct sup.cobounded2*)
qed

end

Again we inherit results from the itering hierarchy.

sublocale *strong-left-kleene-algebra* < *star*: *itering-1* **where** *circ* = *star*
apply *unfold-locales*
apply (*simp add*: *star-slide*)
by (*simp add*: *star-simulation-right*)

context *strong-left-kleene-algebra*
begin

lemma *star-right-induct-mult*:
$y * x \leq y \Longrightarrow y * x^\star \leq y$
by (*simp add*: *star-right-induct*)

lemma *star-right-induct-mult-iff*:
$y * x \leq y \longleftrightarrow y * x^\star \leq y$
using *mult-right-isotone order-trans star.circ-increasing star-right-induct-mult*
by *blast*

lemma *star-simulation-right-equal*:
$z * x = y * z \Longrightarrow z * x^\star = y^\star * z$

by (*metis order.eq-iff star-simulation-left star-simulation-right*)

lemma *star-simulation-star*:
$x * y \leq y * x \Longrightarrow x^\star * y^\star \leq y^\star * x^\star$
by (*simp add*: *star-simulation-left star-simulation-right*)

lemma *star-right-induct-equal*:
$z \sqcup y * x = y \Longrightarrow z * x^\star \leq y$
by (*simp add*: *star-right-induct*)

lemma *star-right-induct-mult-equal*:
$y * x = y \Longrightarrow y * x^\star \leq y$
by (*simp add*: *star-right-induct-mult*)

lemma *star-back-loop-least-fixpoint*:
$x * y \sqcup z = x \Longrightarrow z * y^\star \leq x$
by (*simp add*: *sup-commute star-right-induct-equal*)

lemma *star-back-loop-is-least-fixpoint*:
is-least-fixpoint $(\lambda x . x * y \sqcup z)$ $(z * y^\star)$
proof (*unfold is-least-fixpoint-def*, *rule conjI*)
have $(z * y^\star * y \sqcup z) * y \leq z * y^\star * y \sqcup z$
using *le-supI1 mult-left-isotone star.circ-back-loop-prefixpoint* **by** *auto*
hence $z * y^\star \leq z * y^\star * y \sqcup z$
by (*simp add*: *star-right-induct*)
thus $z * y^\star * y \sqcup z = z * y^\star$
using *order.antisym star.circ-back-loop-prefixpoint* **by** *auto*
next
show $\forall x. \; x * y \sqcup z = x \longrightarrow z * y^\star \leq x$
by (*simp add*: *star-back-loop-least-fixpoint*)
qed

lemma *star-back-loop-mu*:
μ $(\lambda x . x * y \sqcup z) = z * y^\star$
by (*metis least-fixpoint-same star-back-loop-is-least-fixpoint*)

lemma *star-square*:
$x^\star = (1 \sqcup x) * (x * x)^\star$
proof −
let $?f = \lambda y . y * x \sqcup 1$
have *1*: *isotone ?f*
by (*metis sup-left-isotone isotone-def mult-left-isotone*)
have $?f \circ ?f = (\lambda y . y * (x * x) \sqcup (1 \sqcup x))$
by (*simp add*: *sup-assoc sup-commute mult-assoc mult-right-dist-sup o-def*)
thus *?thesis*
using *1* **by** (*metis mu-square mult-left-one star-back-loop-mu*
has-least-fixpoint-def star-back-loop-is-least-fixpoint)
qed

lemma *star-square-2*:
$x^\star = (x * x)^\star * (x \sqcup 1)$
proof −
have $(1 \sqcup x) * (x * x)^\star = (x * x)^\star * 1 \sqcup x * (x * x)^\star$
using *mult-right-dist-sup* **by** *force*
thus *?thesis*
by (*metis* (*no-types*) *order.antisym mult-left-sub-dist-sup star.circ-square-2 star-slide sup-commute star-square*)
qed

lemma *star-circ-simulate-right-plus*:
assumes $z * x \leq y * y^\star * z \sqcup w$
shows $z * x^\star \leq y^\star * (z \sqcup w * x^\star)$
proof −
have $(z \sqcup w * x^\star) * x \leq z * x \sqcup w * x^\star$
using *mult-right-dist-sup star.circ-back-loop-prefixpoint sup-right-isotone* **by** *auto*
also have ... $\leq y * y^\star * z \sqcup w \sqcup w * x^\star$
using *assms sup-left-isotone* **by** *blast*
also have ... $\leq y * y^\star * z \sqcup w * x^\star$
using *le-supI1 star.circ-back-loop-prefixpoint sup-commute* **by** *auto*
also have ... $\leq y^\star * (z \sqcup w * x^\star)$
by (*metis sup.bounded-iff mult-isotone mult-left-isotone mult-left-one mult-left-sub-dist-sup-left star.circ-reflexive star.left-plus-below-circ*)
finally have $y^\star * (z \sqcup w * x^\star) * x \leq y^\star * (z \sqcup w * x^\star)$
by (*metis mult-assoc mult-right-isotone star.circ-transitive-equal*)
thus *?thesis*
by (*metis sup.bounded-iff star-right-induct mult-left-sub-dist-sup-left star.circ-loop-fixpoint*)
qed

lemma *transitive-star*:
$x * x \leq x \Longrightarrow x^\star = 1 \sqcup x$
by (*metis order.antisym star.circ-mult-increasing-2 star.circ-plus-same star-left-induct-mult star-left-unfold-equal*)

lemma *star-sup-2*:
assumes $x * x \leq x$
and $x * y \leq x$
shows $(x \sqcup y)^\star = y^\star * (x \sqcup 1)$
proof −
have $(x \sqcup y)^\star = y^\star * (x * y^\star)^\star$
by (*simp add*: *star.circ-decompose-6 star-sup-1*)
also have ... $= y^\star * x^\star$
using *assms*(*2*) *dual-order.antisym star.circ-back-loop-prefixpoint star-right-induct-mult* **by** *fastforce*
also have ... $= y^\star * (x \sqcup 1)$
by (*simp add*: *assms*(*1*) *sup-commute transitive-star*)
finally show *?thesis*

.
qed

end

The following class contains a generalisation of Kleene algebras, which lacks the right zero axiom.

class *left-zero-kleene-algebra* = *idempotent-left-zero-semiring* + *strong-left-kleene-algebra*
begin

lemma *star-star-absorb*:
$y^\star * (y^\star * x)^\star * y^\star = (y^\star * x)^\star * y^\star$
by (*metis star.circ-transitive-equal star-slide mult-assoc*)

lemma *star-circ-simulate-left-plus*:
assumes $x * z \leq z * y^\star \sqcup w$
shows $x^\star * z \leq (z \sqcup x^\star * w) * y^\star$
proof –
have $x * (x^\star * (w * y^\star)) \leq x^\star * (w * y^\star)$
by (*metis* (*no-types*) *mult-right-sub-dist-sup-left star.circ-loop-fixpoint mult-assoc*)
hence $x * ((z \sqcup x^\star * w) * y^\star) \leq x * z * y^\star \sqcup x^\star * w * y^\star$
using *mult-left-dist-sup mult-right-dist-sup sup-right-isotone mult-assoc* **by** *presburger*
also have ... $\leq (z * y^\star \sqcup w) * y^\star \sqcup x^\star * w * y^\star$
using *assms mult-isotone semiring.add-right-mono* **by** *blast*
also have ... $= z * y^\star \sqcup w * y^\star \sqcup x^\star * w * y^\star$
by (*simp add*: *mult-right-dist-sup star.circ-transitive-equal mult-assoc*)
also have ... $= (z \sqcup w \sqcup x^\star * w) * y^\star$
by (*simp add*: *mult-right-dist-sup*)
also have ... $= (z \sqcup x^\star * w) * y^\star$
by (*metis sup-assoc sup-ge2 le-iff-sup star.circ-loop-fixpoint*)
finally show *?thesis*
by (*metis sup.bounded-iff mult-left-sub-dist-sup-left mult-1-right star.circ-right-unfold-1 star-left-induct*)
qed

lemma *star-one-sup-below*:
$x * y^\star * (1 \sqcup z) \leq x * (y \sqcup z)^\star$
proof –
have $y^\star * z \leq (y \sqcup z)^\star$
using *sup-ge2 order-trans star.circ-increasing star.circ-mult-upper-bound star.circ-sub-dist* **by** *blast*
hence $y^\star \sqcup y^\star * z \leq (y \sqcup z)^\star$
by (*simp add*: *star.circ-sup-upper-bound star.circ-sub-dist*)
hence $y^\star * (1 \sqcup z) \leq (y \sqcup z)^\star$

by (*simp add*: *mult-left-dist-sup*)
thus *?thesis*
by (*metis mult-right-isotone mult-assoc*)
qed

The following theorem is similar to the puzzle where persons insert themselves always in the middle between two groups of people in a line. Here, however, items in the middle annihilate each other, leaving just one group of items behind.

lemma *cancel-separate*:
assumes $x * y \leq 1$
shows $x^\star * y^\star \leq x^\star \sqcup y^\star$
proof −
have $x * y^\star = x \sqcup x * y * y^\star$
by (*metis mult-assoc mult-left-dist-sup mult-1-right star-left-unfold-equal*)
also have ... $\leq x \sqcup y^\star$
by (*meson assms dual-order.trans order.refl star.circ-mult-upper-bound star.circ-reflexive sup-right-isotone*)
also have ... $\leq x^\star \sqcup y^\star$
using *star.circ-increasing sup-left-isotone* **by** *auto*
finally have *1*: $x * y^\star \leq x^\star \sqcup y^\star$
.
have $x * (x^\star \sqcup y^\star) = x * x^\star \sqcup x * y^\star$
by (*simp add*: *mult-left-dist-sup*)
also have ... $\leq x^\star \sqcup y^\star$
using *1* **by** (*metis sup.bounded-iff sup-ge1 order-trans star.left-plus-below-circ*)
finally have *2*: $x * (x^\star \sqcup y^\star) \leq x^\star \sqcup y^\star$
.
have $y^\star \leq x^\star \sqcup y^\star$
by *simp*
hence $y^\star \sqcup x * (x^\star \sqcup y^\star) \leq x^\star \sqcup y^\star$
using *2 sup.bounded-iff* **by** *blast*
thus *?thesis*
by (*metis star-left-induct*)
qed

lemma *star-separate*:
assumes $x * y = bot$
and $y * y = bot$
shows $(x \sqcup y)^\star = x^\star \sqcup y * x^\star$
proof −
have *1*: $y^\star = 1 \sqcup y$
using *assms*(*2*) **by** (*simp add*: *transitive-star*)
have $(x \sqcup y)^\star = y^\star * (x * y^\star)^\star$
by (*simp add*: *star.circ-decompose-6 star-sup-1*)
also have ... $= y^\star * (x * (1 \sqcup y * y^\star))^\star$
by (*simp add*: *star-left-unfold-equal*)
also have ... $= (1 \sqcup y) * x^\star$
using *1* **by** (*simp add*: *assms mult-left-dist-sup*)

also have ... $= x^\star \sqcup y * x^\star$
by (*simp add*: *mult-right-dist-sup*)
finally show *?thesis*
.
qed

end

We can now inherit from the strongest variant of iterings.

sublocale *left-zero-kleene-algebra* < *star*: *itering* **where** *circ* = *star*
apply *unfold-locales*
apply (*metis star.circ-sup-9*)
apply (*metis star.circ-mult-1*)
apply (*simp add*: *star-circ-simulate-right-plus*)
by (*simp add*: *star-circ-simulate-left-plus*)

context *left-zero-kleene-algebra*
begin

lemma *star-absorb*:
$x * y = bot \Longrightarrow x * y^\star = x$
by (*metis sup.bounded-iff antisym-conv star.circ-back-loop-prefixpoint star.circ-elimination*)

lemma *star-separate-2*:
assumes $x * z^+ * y = bot$
and $y * z^+ * y = bot$
and $z * x = bot$
shows $(x^\star \sqcup y * x^\star) * (z * (1 \sqcup y * x^\star))^\star = z^\star * (x^\star \sqcup y * x^\star) * z^\star$
proof –
have *1*: $x^\star * z^+ * y = z^+ * y$
by (*metis assms mult-assoc mult-1-left mult-left-zero star.circ-zero star-simulation-right-equal*)
have *2*: $z^\star * (x^\star \sqcup y * x^\star) * z^+ \leq z^\star * (x^\star \sqcup y * x^\star) * z^\star$
by (*simp add*: *mult-right-isotone star.left-plus-below-circ*)
have $z^\star * z^+ * y * x^\star \leq z^\star * y * x^\star$
by (*metis mult-left-isotone star.left-plus-below-circ star.right-plus-circ star-plus*)
also have ... $\leq z^\star * (x^\star \sqcup y * x^\star)$
by (*simp add*: *mult-assoc mult-left-sub-dist-sup-right*)
also have ... $\leq z^\star * (x^\star \sqcup y * x^\star) * z^\star$
using *sup-right-divisibility star.circ-back-loop-fixpoint* **by** *blast*
finally have *3*: $z^\star * z^+ * y * x^\star \leq z^\star * (x^\star \sqcup y * x^\star) * z^\star$
.
have $z^\star * (x^\star \sqcup y * x^\star) * z^\star * (z * (1 \sqcup y * x^\star)) = z^\star * (x^\star \sqcup y * x^\star) * z^+ \sqcup z^\star * (x^\star \sqcup y * x^\star) * z^+ * y * x^\star$
by (*metis mult-1-right semiring.distrib-left star.circ-plus-same mult-assoc*)
also have ... $= z^\star * (x^\star \sqcup y * x^\star) * z^+ \sqcup z^\star * (1 \sqcup y) * x^\star * z^+ * y * x^\star$
by (*simp add*: *semiring.distrib-right mult-assoc*)

also have ... $= z^\star * (x^\star \sqcup y * x^\star) * z^+ \sqcup z^\star * (1 \sqcup y) * z^+ * y * x^\star$
using *1* **by** (*simp add*: *mult-assoc*)
also have ... $= z^\star * (x^\star \sqcup y * x^\star) * z^+ \sqcup z^\star * z^+ * y * x^\star \sqcup z^\star * y * z^+ * y * x^\star$
using *mult-left-dist-sup mult-right-dist-sup sup-assoc* **by** *auto*
also have ... $= z^\star * (x^\star \sqcup y * x^\star) * z^+ \sqcup z^\star * z^+ * y * x^\star$
by (*metis assms*(*2*) *mult-left-dist-sup mult-left-zero sup-commute sup-monoid.add-0-left mult-assoc*)
also have ... $\le z^\star * (x^\star \sqcup y * x^\star) * z^\star$
using *2 3* **by** *simp*
finally have $(x^\star \sqcup y * x^\star) \sqcup z^\star * (x^\star \sqcup y * x^\star) * z^\star * (z * (1 \sqcup y * x^\star)) \le z^\star * (x^\star \sqcup y * x^\star) * z^\star$
by (*simp add*: *star-outer-increasing*)
hence *4*: $(x^\star \sqcup y * x^\star) * (z * (1 \sqcup y * x^\star))^\star \le z^\star * (x^\star \sqcup y * x^\star) * z^\star$
by (*simp add*: *star-right-induct*)
have *5*: $(x^\star \sqcup y * x^\star) * z^\star \le (x^\star \sqcup y * x^\star) * (z * (1 \sqcup y * x^\star))^\star$
by (*metis sup-ge1 mult-right-isotone mult-1-right star-isotone*)
have $z * (x^\star \sqcup y * x^\star) = z * x^\star \sqcup z * y * x^\star$
by (*simp add*: *mult-assoc mult-left-dist-sup*)
also have ... $= z \sqcup z * y * x^\star$
by (*simp add*: *assms star-absorb*)
also have ... $= z * (1 \sqcup y * x^\star)$
by (*simp add*: *mult-assoc mult-left-dist-sup*)
also have ... $\le (z * (1 \sqcup y * x^\star))^\star$
by (*simp add*: *star.circ-increasing*)
also have ... $\le (x^\star \sqcup y * x^\star) * (z * (1 \sqcup y * x^\star))^\star$
by (*metis le-supE mult-right-sub-dist-sup-left star.circ-loop-fixpoint*)
finally have $z * (x^\star \sqcup y * x^\star) \le (x^\star \sqcup y * x^\star) * (z * (1 \sqcup y * x^\star))^\star$
.
hence $z * (x^\star \sqcup y * x^\star) * (z * (1 \sqcup y * x^\star))^\star \le (x^\star \sqcup y * x^\star) * (z * (1 \sqcup y * x^\star))^\star$
by (*metis mult-assoc mult-left-isotone star.circ-transitive-equal*)
hence $z^\star * (x^\star \sqcup y * x^\star) * z^\star \le (x^\star \sqcup y * x^\star) * (z * (1 \sqcup y * x^\star))^\star$
using *5* **by** (*metis star-left-induct sup.bounded-iff mult-assoc*)
thus *?thesis*
using *4* **by** (*simp add*: *antisym*)
qed

lemma *cancel-separate-eq*:
$x * y \le 1 \Longrightarrow x^\star * y^\star = x^\star \sqcup y^\star$
by (*metis order.antisym cancel-separate star.circ-plus-one star.circ-sup-sub-sup-one-1 star-involutive*)

lemma *cancel-separate-1*:
assumes $x * y \le 1$
shows $(x \sqcup y)^\star = y^\star * x^\star$
proof −
have $y^\star * x^\star * y = y^\star * x^\star * x * y \sqcup y^\star * y$
by (*metis mult-right-dist-sup star.circ-back-loop-fixpoint*)

also have ... $\leq y^\star * x^\star \sqcup y^\star * y$
by (*metis assms semiring.add-right-mono mult-right-isotone mult-1-right mult-assoc*)
also have ... $\leq y^\star * x^\star \sqcup y^\star$
using *semiring.add-left-mono star.right-plus-below-circ* **by** *simp*
also have ... $= y^\star * x^\star$
by (*metis star.circ-back-loop-fixpoint sup.left-idem sup-commute*)
finally have $y^\star * x^\star * y \leq y^\star * x^\star$
by *simp*
hence $y^\star * x^\star * (x \sqcup y) \leq y^\star * x^\star * x \sqcup y^\star * x^\star$
using *mult-left-dist-sup order-lesseq-imp* **by** *fastforce*
also have ... $= y^\star * x^\star$
by (*metis star.circ-back-loop-fixpoint sup.left-idem*)
finally have $(x \sqcup y)^\star \leq y^\star * x^\star$
by (*metis star.circ-decompose-7 star-right-induct-mult sup-commute*)
thus *?thesis*
using *order.antisym star.circ-sub-dist-3 sup-commute* **by** *fastforce*
qed

lemma *plus-sup*:
$(x \sqcup y)^+ = (x^\star * y)^\star * x^+ \sqcup (x^\star * y)^+$
by (*metis semiring.distrib-left star.circ-sup-9 star-plus mult-assoc*)

lemma *plus-half-bot*:
$x * y * x = bot \Longrightarrow (x * y)^+ = x * y$
by (*metis star-absorb star-slide mult-assoc*)

lemma *cancel-separate-1-sup*:
assumes $x * y \leq 1$
and $y * x \leq 1$
shows $(x \sqcup y)^\star = x^\star \sqcup y^\star$
by (*simp add*: *assms cancel-separate-1 cancel-separate-eq sup-commute*)

Lemma *star-separate-3* was contributed by Nicolas Robinson-O'Brien.

lemma *star-separate-3*:
assumes $y * x^\star * y \leq y$
shows $(x \sqcup y)^\star = x^\star \sqcup x^\star * y * x^\star$
proof (*rule order.antisym*)
have $x^\star * y * (x^\star * y)^\star * x^\star \leq x^\star * y * x^\star$
by (*metis assms mult-left-isotone mult-right-isotone star-right-induct-mult mult-assoc*)
thus $(x \sqcup y)^\star \leq x^\star \sqcup x^\star * y * x^\star$
by (*metis order.antisym semiring.add-left-mono star.circ-sup-2 star.circ-sup-sub star.circ-unfold-sum star-decompose-3 star-slide mult-assoc*)
next
show $x^\star \sqcup x^\star * y * x^\star \leq (x \sqcup y)^\star$
using *mult-isotone star.circ-increasing star.circ-sub-dist star.circ-sup-9* **by** *auto*
qed

end

A Kleene algebra is obtained by requiring an idempotent semiring.

class *kleene-algebra* = *left-zero-kleene-algebra* + *idempotent-semiring*

The following classes are variants of Kleene algebras expanded by an additional iteration operation. This is useful to study the Kleene star in computation models that do not use least fixpoints in the refinement order as the semantics of recursion.

class *left-kleene-conway-semiring* = *left-kleene-algebra* + *left-conway-semiring*
begin

lemma *star-below-circ*:
$x^\star \leq x^\circ$
by (*metis circ-left-unfold mult-1-right order-refl star-left-induct*)

lemma *star-zero-below-circ-mult*:
$x^\star * bot \leq x^\circ * y$
by (*simp add*: *mult-isotone star-below-circ*)

lemma *star-mult-circ*:
$x^\star * x^\circ = x^\circ$
by (*metis sup-right-divisibility order.antisym circ-left-unfold star-left-induct-mult star.circ-loop-fixpoint*)

lemma *circ-mult-star*:
$x^\circ * x^\star = x^\circ$
by (*metis sup-assoc sup.bounded-iff circ-left-unfold circ-rtc-2 order.eq-iff left-plus-circ star.circ-sup-sub star.circ-back-loop-prefixpoint star.circ-increasing star-below-circ star-mult-circ star-sup-one*)

lemma *circ-star*:
$x^{\circ\star} = x^\circ$
by (*metis order.antisym circ-reflexive circ-transitive-equal star.circ-increasing star.circ-sup-one-right-unfold star-left-induct-mult-equal*)

lemma *star-circ*:
$x^{\star\circ} = x^{\circ\circ}$
by (*metis order.antisym circ-circ-sup circ-sub-dist le-iff-sup star.circ-rtc-2 star-below-circ*)

lemma *circ-sup-3*:
$(x^\circ * y^\circ)^\star \leq (x \sqcup y)^\circ$
using *circ-star circ-sub-dist-3 star-isotone* **by** *fastforce*

end

class *left-zero-kleene-conway-semiring* = *left-zero-kleene-algebra* + *itering*

begin

subclass *left-kleene-conway-semiring* **..**

lemma *circ-isolate*:
$x^{\circ} = x^{\circ} * bot \sqcup x^{\star}$
by (*metis sup-commute order.antisym circ-sup-upper-bound circ-mult-star circ-simulate-absorb star.left-plus-below-circ star-below-circ zero-right-mult-decreasing*)

lemma *circ-isolate-mult*:
$x^{\circ} * y = x^{\circ} * bot \sqcup x^{\star} * y$
by (*metis circ-isolate mult-assoc mult-left-zero mult-right-dist-sup*)

lemma *circ-isolate-mult-sub*:
$x^{\circ} * y \leq x^{\circ} \sqcup x^{\star} * y$
by (*metis sup-left-isotone circ-isolate-mult zero-right-mult-decreasing*)

lemma *circ-sub-decompose*:
$(x^{\circ} * y)^{\circ} \leq (x^{\star} * y)^{\circ} * x^{\circ}$
proof −
have $x^{\star} * y \sqcup x^{\circ} * bot = x^{\circ} * y$
by (*metis sup.commute circ-isolate-mult*)
hence $(x^{\star} * y)^{\circ} * x^{\circ} = ((x^{\circ} * y)^{\circ} \sqcup x^{\circ})^{\star}$
by (*metis circ-star circ-sup-9 circ-sup-mult-zero star-decompose-1*)
thus *?thesis*
by (*metis circ-star le-iff-sup star.circ-decompose-7 star.circ-unfold-sum*)
qed

lemma *circ-sup-4*:
$(x \sqcup y)^{\circ} = (x^{\star} * y)^{\circ} * x^{\circ}$
apply (*rule order.antisym*)
apply (*metis circ-sup circ-sub-decompose circ-transitive-equal mult-assoc mult-left-isotone*)
by (*metis circ-sup circ-isotone mult-left-isotone star-below-circ*)

lemma *circ-sup-5*:
$(x^{\circ} * y)^{\circ} * x^{\circ} = (x^{\star} * y)^{\circ} * x^{\circ}$
using *circ-sup-4 circ-sup-9* **by** *auto*

lemma *plus-circ*:
$(x^{\star} * x)^{\circ} = x^{\circ}$
by (*metis sup-idem circ-sup-4 circ-decompose-7 circ-star star.circ-decompose-5 star.right-plus-circ*)

end

The following classes add a greatest element.

class *bounded-left-kleene-algebra* = *bounded-idempotent-left-semiring* + *left-kleene-algebra*

sublocale *bounded-left-kleene-algebra* < *star*: *bounded-left-conway-semiring* **where** *circ* = *star* **..**

class *bounded-left-zero-kleene-algebra* = *bounded-idempotent-left-semiring* + *left-zero-kleene-algebra*

sublocale *bounded-left-zero-kleene-algebra* < *star*: *bounded-itering* **where** *circ* = *star* **..**

class *bounded-kleene-algebra* = *bounded-idempotent-semiring* + *kleene-algebra*

sublocale *bounded-kleene-algebra* < *star*: *bounded-itering* **where** *circ* = *star* **..**

We conclude with an alternative axiomatisation of Kleene algebras.

class *kleene-algebra-var* = *idempotent-semiring* + *star* +
  assumes *star-left-unfold-var* : $1 \sqcup y * y^\star \leq y^\star$
  assumes *star-left-induct-var* : $y * x \leq x \longrightarrow y^\star * x \leq x$
  assumes *star-right-induct-var* : $x * y \leq x \longrightarrow x * y^\star \leq x$
begin

subclass *kleene-algebra*
  apply *unfold-locales*
  apply (*rule star-left-unfold-var*)
  apply (*meson sup.bounded-iff mult-right-isotone order-trans star-left-induct-var*)
  by (*meson sup.bounded-iff mult-left-isotone order-trans star-right-induct-var*)

end

end

4 Kleene Relation Algebras

This theory combines Kleene algebras with Stone relation algebras. Relation algebras with transitive closure have been studied by [16]. The weakening to Stone relation algebras allows us to talk about reachability in weighted graphs, for example.

Many results in this theory are used in the correctness proof of Prim's minimum spanning tree algorithm. In particular, they are concerned with the exchange property, preservation of parts of the invariant and with establishing parts of the postcondition.

theory *Kleene-Relation-Algebras*

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

begin

We first note that bounded distributive lattices can be expanded to Kleene algebras by reusing some of the operations.

sublocale *bounded-distrib-lattice* $<$ *comp-inf*: *bounded-kleene-algebra* **where** *star* $= \lambda x\ .\ top$ **and** *one* $=$ *top* **and** *times* $=$ *inf*
  apply *unfold-locales*
  apply (*simp add*: *inf.assoc*)
  apply *simp*
  apply *simp*
  apply (*simp add*: *le-infI2*)
  apply (*simp add*: *inf-sup-distrib2*)
  apply *simp*
  apply *simp*
  apply *simp*
  apply *simp*
  apply *simp*
  apply (*simp add*: *inf-sup-distrib1*)
  apply *simp*
  apply *simp*
  by (*simp add*: *inf-assoc*)

We add the Kleene star operation to each of bounded distributive allegories, pseudocomplemented distributive allegories and Stone relation algebras. We start with single-object bounded distributive allegories.

class *bounded-distrib-kleene-allegory* $=$ *bounded-distrib-allegory* $+$ *kleene-algebra*
begin

subclass *bounded-kleene-algebra* **..**

lemma *conv-star-conv*:
  $x^\star \le x^{T\star T}$
proof $-$
  have $x^{T\star} * x^T \le x^{T\star}$
    by (*simp add*: *star.right-plus-below-circ*)
  hence *1*: $x * x^{T\star T} \le x^{T\star T}$
    using *conv-dist-comp conv-isotone* **by** *fastforce*
  have $1 \le x^{T\star T}$
    by (*simp add*: *reflexive-conv-closed star.circ-reflexive*)
  hence $1 \sqcup x * x^{T\star T} \le x^{T\star T}$
    using *1* **by** *simp*
  thus *?thesis*
    using *star-left-induct* **by** *fastforce*
qed

It follows that star and converse commute.

lemma *conv-star-commute*:
  $x^{\star T} = x^{T\star}$
proof (*rule order.antisym*)

show $x^{\star T} \leq x^{T\star}$
using *conv-star-conv conv-isotone* **by** *fastforce*
next
show $x^{T\star} \leq x^{\star T}$
by (*metis conv-star-conv conv-involutive*)
qed

lemma *conv-plus-commute*:
$x^{+T} = x^{T+}$
by (*simp add*: *conv-dist-comp conv-star-commute star-plus*)

Lemma *reflexive-inf-star* was contributed by Nicolas Robinson-O'Brien.

lemma *reflexive-inf-star*:
assumes *reflexive* y
shows $y \sqcap x^\star = 1 \sqcup (y \sqcap x^+)$
by (*simp add*: *assms star-left-unfold-equal sup.absorb2 sup-inf-distrib1*)

The following results are variants of a separation lemma of Kleene algebras.

lemma *cancel-separate-2*:
assumes $x * y \leq 1$
shows $((w \sqcap x) \sqcup (z \sqcap y))^\star = (z \sqcap y)^\star * (w \sqcap x)^\star$
proof –
have $(w \sqcap x) * (z \sqcap y) \leq 1$
by (*meson assms comp-isotone order.trans inf.cobounded2*)
thus *?thesis*
using *cancel-separate-1 sup-commute* **by** *simp*
qed

lemma *cancel-separate-3*:
assumes $x * y \leq 1$
shows $(w \sqcap x)^\star * (z \sqcap y)^\star = (w \sqcap x)^\star \sqcup (z \sqcap y)^\star$
proof –
have $(w \sqcap x) * (z \sqcap y) \leq 1$
by (*meson assms comp-isotone order.trans inf.cobounded2*)
thus *?thesis*
by (*simp add*: *cancel-separate-eq*)
qed

lemma *cancel-separate-4*:
assumes $z * y \leq 1$
and $w \leq y \sqcup z$
and $x \leq y \sqcup z$
shows $w^\star * x^\star = (w \sqcap y)^\star * ((w \sqcap z)^\star \sqcup (x \sqcap y)^\star) * (x \sqcap z)^\star$
proof –
have $w^\star * x^\star = ((w \sqcap y) \sqcup (w \sqcap z))^\star * ((x \sqcap y) \sqcup (x \sqcap z))^\star$
by (*metis assms(2,3) inf.orderE inf-sup-distrib1*)
also have ... $= (w \sqcap y)^\star * ((w \sqcap z)^\star * (x \sqcap y)^\star) * (x \sqcap z)^\star$
by (*metis assms(1) cancel-separate-2 sup-commute mult-assoc*)

finally show *?thesis*
  by (*simp add*: *assms*(*1*) *cancel-separate-3*)
qed

lemma *cancel-separate-5*:
  assumes $w * z^T \leq 1$
  shows $w \sqcap x * (y \sqcap z) \leq y$
proof −
  have $w \sqcap x * (y \sqcap z) \leq (x \sqcap w * (y \sqcap z)^T) * (y \sqcap z)$
    by (*metis dedekind-2 inf-commute*)
  also have ... $\leq w * z^T * (y \sqcap z)$
    by (*simp add*: *conv-dist-inf inf.coboundedI2 mult-left-isotone mult-right-isotone*)
  also have ... $\leq y \sqcap z$
    by (*metis assms mult-1-left mult-left-isotone*)
  finally show *?thesis*
    by *simp*
qed

lemma *cancel-separate-6*:
  assumes $z * y \leq 1$
    and $w \leq y \sqcup z$
    and $x \leq y \sqcup z$
    and $v * z^T \leq 1$
    and $v \sqcap y^\star = bot$
  shows $v \sqcap w^\star * x^\star \leq x \sqcup w$
proof −
  have $v \sqcap (w \sqcap y)^\star * (x \sqcap y)^\star \leq v \sqcap y^\star * (x \sqcap y)^\star$
    using *comp-inf.mult-right-isotone mult-left-isotone star-isotone* **by** *simp*
  also have ... $\leq v \sqcap y^\star$
    by (*simp add*: *inf.coboundedI2 star.circ-increasing star.circ-mult-upper-bound star-right-induct-mult*)
  finally have *1*: $v \sqcap (w \sqcap y)^\star * (x \sqcap y)^\star = bot$
    using *assms*(*5*) *le-bot* **by** *simp*
  have $v \sqcap w^\star * x^\star = v \sqcap (w \sqcap y)^\star * ((w \sqcap z)^\star \sqcup (x \sqcap y)^\star) * (x \sqcap z)^\star$
    using *assms*(*1*−*3*) *cancel-separate-4* **by** *simp*
  also have ... $= (v \sqcap (w \sqcap y)^\star * ((w \sqcap z)^\star \sqcup (x \sqcap y)^\star) * (x \sqcap z)^\star * (x \sqcap z)) \sqcup (v \sqcap (w \sqcap y)^\star * ((w \sqcap z)^\star \sqcup (x \sqcap y)^\star))$
    by (*metis inf-sup-distrib1 star.circ-back-loop-fixpoint*)
  also have ... $\leq x \sqcup (v \sqcap (w \sqcap y)^\star * ((w \sqcap z)^\star \sqcup (x \sqcap y)^\star))$
    using *assms*(*4*) *cancel-separate-5 semiring.add-right-mono* **by** *simp*
  also have ... $= x \sqcup (v \sqcap (w \sqcap y)^\star * (w \sqcap z)^\star)$
    using *1* **by** (*simp add*: *inf-sup-distrib1 mult-left-dist-sup sup-monoid.add-assoc*)
  also have ... $= x \sqcup (v \sqcap (w \sqcap y)^\star * (w \sqcap z)^\star * (w \sqcap z)) \sqcup (v \sqcap (w \sqcap y)^\star)$
    by (*metis comp-inf.semiring.distrib-left star.circ-back-loop-fixpoint sup-assoc*)
  also have ... $\leq x \sqcup w \sqcup (v \sqcap (w \sqcap y)^\star)$
    using *assms*(*4*) *cancel-separate-5 sup-left-isotone sup-right-isotone* **by** *simp*
  also have ... $\leq x \sqcup w \sqcup (v \sqcap y^\star)$

using *comp-inf.mult-right-isotone star-isotone sup-right-isotone* **by** *simp*
finally show *?thesis*
using *assms*(*5*) *le-bot* **by** *simp*
qed

We show several results about the interaction of vectors and the Kleene star.

lemma *vector-star-1*:
assumes *vector x*
shows $x^T * (x * x^T)^\star \leq x^T$
proof −
have $x^T * (x * x^T)^\star = (x^T * x)^\star * x^T$
by (*simp add*: *star-slide*)
also have ... $\leq$ *top* $* x^T$
by (*simp add*: *mult-left-isotone*)
also have ... $= x^T$
using *assms vector-conv-covector* **by** *auto*
finally show *?thesis*
.
qed

lemma *vector-star-2*:
vector x $\Longrightarrow x^T * (x * x^T)^\star \leq x^T * bot^\star$
by (*simp add*: *star-absorb vector-star-1*)

lemma *vector-vector-star*:
vector v $\Longrightarrow (v * v^T)^\star = 1 \sqcup v * v^T$
by (*simp add*: *transitive-star vv-transitive*)

lemma *equivalence-star-closed*:
equivalence x $\Longrightarrow$ *equivalence* $(x^\star)$
by (*simp add*: *conv-star-commute star.circ-reflexive star.circ-transitive-equal*)

lemma *equivalence-plus-closed*:
equivalence x $\Longrightarrow$ *equivalence* (x^+)
by (*simp add*: *conv-star-commute star.circ-reflexive star.circ-sup-one-left-unfold star.circ-transitive-equal*)

The following equivalence relation characterises the component trees of a forest. This is a special case of undirected reachability in a directed graph.

abbreviation *forest-components f* $\equiv f^{T\star} * f^\star$

lemma *forest-components-equivalence*:
injective x $\Longrightarrow$ *equivalence* (*forest-components x*)
apply (*intro conjI*)
apply (*simp add*: *reflexive-mult-closed star.circ-reflexive*)
apply (*metis cancel-separate-1 order.eq-iff star.circ-transitive-equal*)
by (*simp add*: *conv-dist-comp conv-star-commute*)

lemma *forest-components-increasing*:
$x \le$ *forest-components* x
by (*metis order.trans mult-left-isotone mult-left-one star.circ-increasing star.circ-reflexive*)

lemma *forest-components-isotone*:
$x \le y \Longrightarrow$ *forest-components* $x \le$ *forest-components* y
by (*simp add*: *comp-isotone conv-isotone star-isotone*)

lemma *forest-components-idempotent*:
injective $x \Longrightarrow$ *forest-components* (*forest-components* x) $=$ *forest-components* x
by (*metis forest-components-equivalence cancel-separate-1 star.circ-transitive-equal star-involutive*)

lemma *forest-components-star*:
injective $x \Longrightarrow$ (*forest-components* x)* $=$ *forest-components* x
using *forest-components-equivalence forest-components-idempotent star.circ-transitive-equal* **by** *simp*

The following lemma shows that the nodes reachable in the graph can be reached by only using edges between reachable nodes.

lemma *reachable-restrict*:
assumes *vector* r
shows $r^T * g^\star = r^T * ((r^T * g^\star)^T * (r^T * g^\star) \sqcap g)^\star$
proof $-$
have *1*: $r^T \le r^T * ((r^T * g^\star)^T * (r^T * g^\star) \sqcap g)^\star$
using *mult-right-isotone mult-1-right star.circ-reflexive* **by** *fastforce*
have *2*: *covector* $(r^T * g^\star)$
using *assms covector-mult-closed vector-conv-covector* **by** *auto*
have $r^T * ((r^T * g^\star)^T * (r^T * g^\star) \sqcap g)^\star * g \le r^T * g^\star * g$
by (*simp add*: *mult-left-isotone mult-right-isotone star-isotone*)
also have ... $\le r^T * g^\star$
by (*simp add*: *mult-assoc mult-right-isotone star.left-plus-below-circ star-plus*)
finally have $r^T * ((r^T * g^\star)^T * (r^T * g^\star) \sqcap g)^\star * g = r^T * ((r^T * g^\star)^T * (r^T * g^\star) \sqcap g)^\star * g \sqcap r^T * g^\star$
by (*simp add*: *le-iff-inf*)
also have ... $= r^T * ((r^T * g^\star)^T * (r^T * g^\star) \sqcap g)^\star * (g \sqcap r^T * g^\star)$
using *assms covector-comp-inf covector-mult-closed vector-conv-covector* **by** *auto*
also have ... $= (r^T * ((r^T * g^\star)^T * (r^T * g^\star) \sqcap g)^\star \sqcap r^T * g^\star) * (g \sqcap r^T * g^\star)$
by (*simp add*: *inf.absorb2 inf-commute mult-right-isotone star-isotone*)
also have ... $= r^T * ((r^T * g^\star)^T * (r^T * g^\star) \sqcap g)^\star * (g \sqcap r^T * g^\star \sqcap (r^T * g^\star)^T)$
using *2* **by** (*metis comp-inf-vector-1*)
also have ... $= r^T * ((r^T * g^\star)^T * (r^T * g^\star) \sqcap g)^\star * ((r^T * g^\star)^T \sqcap r^T * g^\star \sqcap g)$
using *inf-commute inf-assoc* **by** *simp*
also have ... $= r^T * ((r^T * g^\star)^T * (r^T * g^\star) \sqcap g)^\star * ((r^T * g^\star)^T * (r^T * g^\star) \sqcap g)$
using *2* **by** (*metis covector-conv-vector inf-top.right-neutral vector-inf-comp*)
also have ... $\le r^T * ((r^T * g^\star)^T * (r^T * g^\star) \sqcap g)^\star$

by (*simp add*: *mult-assoc mult-right-isotone star.left-plus-below-circ star-plus*)
finally have $r^T * g^\star \le r^T * ((r^T * g^\star)^T * (r^T * g^\star) \sqcap g)^\star$
using *1 star-right-induct* **by** *auto*
thus *?thesis*
by (*simp add*: *order.eq-iff mult-right-isotone star-isotone*)
qed

lemma *kruskal-acyclic-inv-1*:
assumes *injective f*
and $e * \mathit{forest\text{-}components}\ f * e = \mathit{bot}$
shows $(f \sqcap \mathit{top} * e * f^{T\star})^T * f^\star * e = \mathit{bot}$
proof −
let $?q = \mathit{top} * e * f^{T\star}$
let $?F = \mathit{forest\text{-}components}\ f$
have $(f \sqcap ?q)^T * f^\star * e = ?q^T \sqcap f^T * f^\star * e$
by (*metis* (*mono-tags*) *comp-associative conv-dist-inf covector-conv-vector inf-vector-comp vector-top-closed*)
also have ... $\le ?q^T \sqcap ?F * e$
using *comp-inf.mult-right-isotone mult-left-isotone star.circ-increasing* **by** *simp*
also have ... $= f^\star * e^T * \mathit{top} \sqcap ?F * e$
by (*simp add*: *conv-dist-comp conv-star-commute mult-assoc*)
also have ... $\le ?F * e^T * \mathit{top} \sqcap ?F * e$
by (*metis conv-dist-comp conv-star-commute conv-top inf.sup-left-isotone star.circ-right-top star-outer-increasing mult-assoc*)
also have ... $= ?F * (e^T * \mathit{top} \sqcap ?F * e)$
by (*metis assms*(*1*) *forest-components-equivalence equivalence-comp-dist-inf mult-assoc*)
also have ... $= (?F \sqcap \mathit{top} * e) * ?F * e$
by (*simp add*: *comp-associative comp-inf-vector-1 conv-dist-comp inf-vector-comp*)
also have ... $\le \mathit{top} * e * ?F * e$
by (*simp add*: *mult-left-isotone*)
also have ... $= \mathit{bot}$
using *assms*(*2*) *mult-assoc* **by** *simp*
finally show *?thesis*
by (*simp add*: *bot-unique*)
qed

lemma *kruskal-forest-components-inf-1*:
assumes $f \le w \sqcup w^T$
and *injective w*
and $f \le \mathit{forest\text{-}components}\ g$
shows $f * \mathit{forest\text{-}components}\ (\mathit{forest\text{-}components}\ g \sqcap w) \le \mathit{forest\text{-}components}\ (\mathit{forest\text{-}components}\ g \sqcap w)$
proof −
let $?f = \mathit{forest\text{-}components}\ g$
let $?w = \mathit{forest\text{-}components}\ (?f \sqcap w)$
have $f * ?w = (f \sqcap (w \sqcup w^T)) * ?w$

by (*simp add*: *assms*(*1*) *inf.absorb1*)
also have ... $= (f \sqcap w) * ?w \sqcup (f \sqcap w^T) * ?w$
by (*simp add*: *inf-sup-distrib1 semiring.distrib-right*)
also have ... $\leq (?f \sqcap w) * ?w \sqcup (f \sqcap w^T) * ?w$
using *assms*(*3*) *inf.sup-left-isotone mult-left-isotone sup-left-isotone* **by** *simp*
also have ... $\leq (?f \sqcap w) * ?w \sqcup (?f \sqcap w^T) * ?w$
using *assms*(*3*) *inf.sup-left-isotone mult-left-isotone sup-right-isotone* **by** *simp*
also have ... $= (?f \sqcap w) * ?w \sqcup (?f \sqcap w)^T * ?w$
by (*simp add*: *conv-dist-comp conv-dist-inf conv-star-commute*)
also have ... $\leq (?f \sqcap w) * ?w \sqcup ?w$
by (*metis star.circ-loop-fixpoint sup-ge1 sup-right-isotone*)
also have ... $= ?w \sqcup (?f \sqcap w) * (?f \sqcap w)^\star \sqcup (?f \sqcap w) * (?f \sqcap w)^{T+} * (?f \sqcap w)^\star$
by (*metis comp-associative mult-left-dist-sup star.circ-loop-fixpoint sup-commute sup-assoc*)
also have ... $\leq ?w \sqcup (?f \sqcap w)^\star \sqcup (?f \sqcap w) * (?f \sqcap w)^{T+} * (?f \sqcap w)^\star$
using *star.left-plus-below-circ sup-left-isotone sup-right-isotone* **by** *auto*
also have ... $= ?w \sqcup (?f \sqcap w) * (?f \sqcap w)^{T+} * (?f \sqcap w)^\star$
by (*metis star.circ-loop-fixpoint sup.right-idem*)
also have ... $\leq ?w \sqcup w * w^T * ?w$
using *comp-associative conv-dist-inf mult-isotone sup-right-isotone* **by** *simp*
also have ... $= ?w$
by (*metis assms*(*2*) *coreflexive-comp-top-inf inf.cobounded2 sup.orderE*)
finally show *?thesis*
by *simp*
qed

lemma *kruskal-forest-components-inf*:
assumes $f \leq w \sqcup w^T$
and *injective* w
shows *forest-components* $f \leq$ *forest-components* (*forest-components* $f \sqcap w$)
proof −
let $?f =$ *forest-components* f
let $?w =$ *forest-components* $(?f \sqcap w)$
have *1*: $1 \leq ?w$
by (*simp add*: *reflexive-mult-closed star.circ-reflexive*)
have $f * ?w \leq ?w$
using *assms forest-components-increasing kruskal-forest-components-inf-1* **by** *simp*
hence *2*: $f^\star \leq ?w$
using *1 star-left-induct* **by** *fastforce*
have $f^T * ?w \leq ?w$
apply (*rule kruskal-forest-components-inf-1*)
apply (*metis assms*(*1*) *conv-dist-sup conv-involutive conv-isotone sup-commute*)
apply (*simp add*: *assms*(*2*))
by (*metis le-supI2 star.circ-back-loop-fixpoint star.circ-increasing*)
thus $?f \leq ?w$
using *2 star-left-induct* **by** *simp*

qed

end

We next add the Kleene star to single-object pseudocomplemented distributive allegories.

class *pd-kleene-allegory* = *pd-allegory* + *bounded-distrib-kleene-allegory*
begin

The following definitions and results concern acyclic graphs and forests.

abbreviation *acyclic* :: *'a* ⇒ *bool* **where** $acyclic\ x \equiv x^+ \leq -1$

abbreviation *forest* :: *'a* ⇒ *bool* **where** $forest\ x \equiv injective\ x \wedge acyclic\ x$

lemma *forest-bot*:
forest bot
by *simp*

lemma *acyclic-down-closed*:
$x \leq y \Longrightarrow acyclic\ y \Longrightarrow acyclic\ x$
using *comp-isotone star-isotone* **by** *fastforce*

lemma *forest-down-closed*:
$x \leq y \Longrightarrow forest\ y \Longrightarrow forest\ x$
using *conv-isotone mult-isotone star-isotone* **by** *fastforce*

lemma *acyclic-star-below-complement*:
$acyclic\ w \longleftrightarrow w^{T\star} \leq -w$
by (*simp add*: *conv-star-commute schroeder-4-p*)

lemma *acyclic-star-below-complement-1*:
$acyclic\ w \longleftrightarrow w^\star \sqcap w^T = bot$
using *pseudo-complement schroeder-5-p* **by** *force*

lemma *acyclic-star-inf-conv*:
assumes *acyclic w*
shows $w^\star \sqcap w^{T\star} = 1$
proof −
have $w^+ \sqcap w^{T\star} \leq (w \sqcap w^{T\star}) * w^\star$
by (*metis conv-star-commute dedekind-2 star.circ-transitive-equal*)
also have ... $= bot$
by (*metis assms conv-star-commute p-antitone-iff pseudo-complement schroeder-4-p semiring.mult-not-zero star.circ-circ-mult star-involutive star-one*)
finally have $w^\star \sqcap w^{T\star} \leq 1$
by (*metis order.eq-iff le-bot mult-left-zero star.circ-plus-one star.circ-zero star-left-unfold-equal sup-inf-distrib1*)
thus *?thesis*
by (*simp add*: *order.antisym star.circ-reflexive*)
qed

lemma *acyclic-asymmetric*:
acyclic $w \Longrightarrow$ *asymmetric* w
by (*simp add*: *dual-order.trans pseudo-complement schroeder-5-p star.circ-increasing*)

lemma *forest-separate*:
assumes *forest* x
shows $x^\star * x^{T\star} \sqcap x^T * x \leq 1$
proof –
have $x^\star * 1 \leq -x^T$
using *assms schroeder-5-p* **by** *force*
hence *1*: $x^\star \sqcap x^T = bot$
by (*simp add*: *pseudo-complement*)
have $x^\star \sqcap x^T * x = (1 \sqcup x^\star * x) \sqcap x^T * x$
using *star.circ-right-unfold-1* **by** *simp*
also have ... $= (1 \sqcap x^T * x) \sqcup (x^\star * x \sqcap x^T * x)$
by (*simp add*: *inf-sup-distrib2*)
also have ... $\leq 1 \sqcup (x^\star * x \sqcap x^T * x)$
using *sup-left-isotone* **by** *simp*
also have ... $= 1 \sqcup (x^\star \sqcap x^T) * x$
by (*simp add*: *assms injective-comp-right-dist-inf*)
also have ... $= 1$
using *1* **by** *simp*
finally have *2*: $x^\star \sqcap x^T * x \leq 1$
.
hence *3*: $x^{T\star} \sqcap x^T * x \leq 1$
by (*metis* (*mono-tags*, *lifting*) *conv-star-commute conv-dist-comp conv-dist-inf conv-involutive coreflexive-symmetric*)
have $x^\star * x^{T\star} \sqcap x^T * x \leq (x^\star \sqcup x^{T\star}) \sqcap x^T * x$
using *assms cancel-separate inf.sup-left-isotone* **by** *simp*
also have ... ≤ 1
using *2 3* **by** (*simp add*: *inf-sup-distrib2*)
finally show *?thesis*
.
qed

The following definition captures the components of undirected weighted graphs.

abbreviation *components* $g \equiv (--g)^\star$

lemma *components-equivalence*:
symmetric $x \Longrightarrow$ *equivalence* (*components* x)
by (*simp add*: *conv-star-commute conv-complement star.circ-reflexive star.circ-transitive-equal*)

lemma *components-increasing*:
$x \leq$ *components* x
using *order-trans pp-increasing star.circ-increasing* **by** *blast*

lemma *components-isotone*:
  $x \leq y \Longrightarrow$ *components* $x \leq$ *components* y
  by (*simp add*: *pp-isotone star-isotone*)

lemma *cut-reachable*:
  assumes $v^T = r^T * t^\star$
    and $t \leq g$
   shows $v * -v^T \sqcap g \leq (r^T * g^\star)^T * (r^T * g^\star)$
proof –
  have $v * -v^T \sqcap g \leq v * top \sqcap g$
   using *inf.sup-left-isotone mult-right-isotone top-greatest* **by** *blast*
  also have ... $= (r^T * t^\star)^T * top \sqcap g$
   by (*metis assms*(*1*) *conv-involutive*)
  also have ... $\leq (r^T * g^\star)^T * top \sqcap g$
   using *assms*(*2*) *conv-isotone inf.sup-left-isotone mult-left-isotone mult-right-isotone star-isotone* **by** *auto*
  also have ... $\leq (r^T * g^\star)^T * ((r^T * g^\star) * g)$
   by (*metis conv-involutive dedekind-1 inf-top.left-neutral*)
  also have ... $\leq (r^T * g^\star)^T * (r^T * g^\star)$
   by (*simp add*: *mult-assoc mult-right-isotone star.left-plus-below-circ star-plus*)
  finally show *?thesis*
   .
qed

The following lemma shows that the predecessors of visited nodes in the minimum spanning tree extending the current tree have all been visited.

lemma *predecessors-reachable*:
  assumes *vector* r
    and *injective* r
    and $v^T = r^T * t^\star$
    and *forest* w
    and $t \leq w$
    and $w \leq (r^T * g^\star)^T * (r^T * g^\star) \sqcap g$
    and $r^T * g^\star \leq r^T * w^\star$
   shows $w * v \leq v$
proof –
  have $w * r \leq (r^T * g^\star)^T * (r^T * g^\star) * r$
   using *assms*(*6*) *mult-left-isotone* **by** *auto*
  also have ... $\leq (r^T * g^\star)^T * top$
   by (*simp add*: *mult-assoc mult-right-isotone*)
  also have ... $= (r^T * g^\star)^T$
   by (*simp add*: *assms*(*1*) *comp-associative conv-dist-comp*)
  also have ... $\leq (r^T * w^\star)^T$
   by (*simp add*: *assms*(*7*) *conv-isotone*)
  also have ... $= w^{T\star} * r$
   by (*simp add*: *conv-dist-comp conv-star-commute*)
  also have ... $\leq -w * r$
   using *assms*(*4*) **by** (*simp add*: *mult-left-isotone*

acyclic-star-below-complement)
  also have ... $\leq -(w * r)$
    by (*simp add*: *assms*(*2*) *comp-injective-below-complement*)
  finally have *1*: $w * r = bot$
    by (*simp add*: *le-iff-inf*)
  have $v = t^{T\star} * r$
    by (*metis assms*(*3*) *conv-dist-comp conv-involutive conv-star-commute*)
  also have ... $= t^T * v \sqcup r$
    by (*simp add*: *calculation star.circ-loop-fixpoint*)
  also have ... $\leq w^T * v \sqcup r$
    using *assms*(*5*) *comp-isotone conv-isotone semiring.add-right-mono* **by** *auto*
  finally have $w * v \leq w * w^T * v \sqcup w * r$
    by (*simp add*: *comp-left-dist-sup mult-assoc mult-right-isotone*)
  also have ... $= w * w^T * v$
    using *1* **by** *simp*
  also have ... $\leq v$
    using *assms*(*4*) **by** (*simp add*: *star-left-induct-mult-iff star-sub-one*)
  finally show *?thesis*
    .
qed

4.1 Prim's Algorithm

The following results are used for proving the correctness of Prim's minimum spanning tree algorithm.

4.1.1 Preservation of Invariant

We first treat the preservation of the invariant. The following lemma shows that the while-loop preserves that v represents the nodes of the constructed tree. The remaining lemmas in this section show that t is a spanning tree. The exchange property is treated in the following two sections.

lemma *reachable-inv*:
  assumes *vector* v
    and $e \leq v * -v^T$
    and $e * t = bot$
    and $v^T = r^T * t^\star$
  shows $(v \sqcup e^T * top)^T = r^T * (t \sqcup e)^\star$
proof –
  have *1*: $v^T \leq r^T * (t \sqcup e)^\star$
    by (*simp add*: *assms*(*4*) *mult-right-isotone star.circ-sub-dist*)
  have *2*: $(e^T * top)^T = top * e$
    by (*simp add*: *conv-dist-comp*)
  also have ... $= top * (v * -v^T \sqcap e)$
    by (*simp add*: *assms*(*2*) *inf-absorb2*)
  also have ... $\leq top * (v * top \sqcap e)$
    using *inf.sup-left-isotone mult-right-isotone top-greatest* **by** *blast*
  also have ... $= top * v^T * e$

by $(simp\ add: comp\text{-}inf\text{-}vector\ inf.sup\text{-}monoid.add\text{-}commute)$
also have ... $= v^T * e$
using $assms(1)$ $vector\text{-}conv\text{-}covector$ by $auto$
also have ... $\leq r^T * (t \sqcup e)^\star * e$
using 1 by $(simp\ add: mult\text{-}left\text{-}isotone)$
also have ... $\leq r^T * (t \sqcup e)^\star * (t \sqcup e)$
by $(simp\ add: mult\text{-}right\text{-}isotone)$
also have ... $\leq r^T * (t \sqcup e)^\star$
by $(simp\ add: comp\text{-}associative\ mult\text{-}right\text{-}isotone\ star.right\text{-}plus\text{-}below\text{-}circ)$
finally have 3: $(v \sqcup e^T * top)^T \leq r^T * (t \sqcup e)^\star$
using 1 by $(simp\ add: conv\text{-}dist\text{-}sup)$
have $r^T \leq r^T * t^\star$
using $sup.bounded\text{-}iff\ star.circ\text{-}back\text{-}loop\text{-}prefixpoint$ by $blast$
also have ... $\leq (v \sqcup e^T * top)^T$
by $(metis\ assms(4)\ conv\text{-}isotone\ sup\text{-}ge1)$
finally have 4: $r^T \leq (v \sqcup e^T * top)^T$
.
have $(v \sqcup e^T * top)^T * (t \sqcup e) = (v \sqcup e^T * top)^T * t \sqcup (v \sqcup e^T * top)^T * e$
by $(simp\ add: mult\text{-}left\text{-}dist\text{-}sup)$
also have ... $\leq (v \sqcup e^T * top)^T * t \sqcup top * e$
using $comp\text{-}isotone\ semiring.add\text{-}left\text{-}mono$ by $auto$
also have ... $= v^T * t \sqcup top * e * t \sqcup top * e$
using 2 by $(simp\ add: conv\text{-}dist\text{-}sup\ mult\text{-}right\text{-}dist\text{-}sup)$
also have ... $= v^T * t \sqcup top * e$
by $(simp\ add: assms(3)\ comp\text{-}associative)$
also have ... $\leq r^T * t^\star \sqcup top * e$
by $(metis\ assms(4)\ star.circ\text{-}back\text{-}loop\text{-}fixpoint\ sup\text{-}ge1\ sup\text{-}left\text{-}isotone)$
also have ... $= v^T \sqcup top * e$
by $(simp\ add: assms(4))$
finally have 5: $(v \sqcup e^T * top)^T * (t \sqcup e) \leq (v \sqcup e^T * top)^T$
using 2 by $(simp\ add: conv\text{-}dist\text{-}sup)$
have $r^T * (t \sqcup e)^\star \leq (v \sqcup e^T * top)^T * (t \sqcup e)^\star$
using 4 by $(simp\ add: mult\text{-}left\text{-}isotone)$
also have ... $\leq (v \sqcup e^T * top)^T$
using 5 by $(simp\ add: star\text{-}right\text{-}induct\text{-}mult)$
finally show $?thesis$
using 3 by $(simp\ add: order.eq\text{-}iff)$
qed

The next result is used to show that the while-loop preserves acyclicity of the constructed tree.

lemma $acyclic\text{-}inv$:
assumes $acyclic\ t$
and $vector\ v$
and $e \leq v * -v^T$
and $t \leq v * v^T$
shows $acyclic\ (t \sqcup e)$
proof −
have $t^+ * e \leq t^+ * v * -v^T$

by $(simp\ add: assms(3)\ comp\text{-}associative\ mult\text{-}right\text{-}isotone)$
also have ... $\leq v * v^T * t^\star * v * -v^T$
by $(simp\ add: assms(4)\ mult\text{-}left\text{-}isotone)$
also have ... $\leq v * top * -v^T$
by $(metis\ mult\text{-}assoc\ mult\text{-}left\text{-}isotone\ mult\text{-}right\text{-}isotone\ top\text{-}greatest)$
also have ... $= v * -v^T$
by $(simp\ add: assms(2))$
also have ... ≤ -1
by $(simp\ add: pp\text{-}increasing\ schroeder\text{-}3\text{-}p)$
finally have $1: t^+ * e \leq -1$
.
have $2: e * t^\star = e$
using $assms(2{-}4)\ et(1)\ star\text{-}absorb$ **by** $blast$
have $e^\star = 1 \sqcup e \sqcup e * e * e^\star$
by $(metis\ star.circ\text{-}loop\text{-}fixpoint\ star\text{-}square\text{-}2\ sup\text{-}commute)$
also have ... $= 1 \sqcup e$
using $assms(2,3)\ ee\ comp\text{-}left\text{-}zero\ bot\text{-}least\ sup\text{-}absorb1$ **by** $simp$
finally have $3: e^\star = 1 \sqcup e$
.
have $e \leq v * -v^T$
by $(simp\ add: assms(3))$
also have ... ≤ -1
by $(simp\ add: pp\text{-}increasing\ schroeder\text{-}3\text{-}p)$
finally have $4: t^+ * e \sqcup e \leq -1$
using 1 **by** $simp$
have $(t \sqcup e)^+ = (t \sqcup e) * t^\star * (e * t^\star)^\star$
using $star\text{-}sup\text{-}1\ mult\text{-}assoc$ **by** $simp$
also have ... $= (t \sqcup e) * t^\star * (1 \sqcup e)$
using $2\ 3$ **by** $simp$
also have ... $= t^+ * (1 \sqcup e) \sqcup e * t^\star * (1 \sqcup e)$
by $(simp\ add: comp\text{-}right\text{-}dist\text{-}sup)$
also have ... $= t^+ * (1 \sqcup e) \sqcup e * (1 \sqcup e)$
using 2 **by** $simp$
also have ... $= t^+ * (1 \sqcup e) \sqcup e$
using 3 **by** $(metis\ star\text{-}absorb\ assms(2,3)\ ee)$
also have ... $= t^+ \sqcup t^+ * e \sqcup e$
by $(simp\ add: mult\text{-}left\text{-}dist\text{-}sup)$
also have ... ≤ -1
using 4 **by** $(metis\ assms(1)\ sup.absorb1\ sup.orderI\ sup\text{-}assoc)$
finally show $?thesis$
.
qed

The following lemma shows that the extended tree is in the component reachable from the root.

lemma $mst\text{-}subgraph\text{-}inv\text{-}2$:
assumes $regular\ (v * v^T)$
and $t \leq v * v^T \sqcap --g$
and $v^T = r^T * t^\star$

and $e \leq v * -v^T \sqcap --g$
and *vector* v
and *regular* $((v \sqcup e^T * top) * (v \sqcup e^T * top)^T)$
shows $t \sqcup e \leq (r^T * (--((v \sqcup e^T * top) * (v \sqcup e^T * top)^T \sqcap g))^\star)^T * (r^T * (--((v \sqcup e^T * top) * (v \sqcup e^T * top)^T \sqcap g))^\star)$
proof $-$
let $?v = v \sqcup e^T * top$
let $?G = ?v * ?v^T \sqcap g$
let $?c = r^T * (--?G)^\star$
have $v^T \leq r^T * (--(v * v^T \sqcap g))^\star$
using *assms*$(1-3)$ *inf-pp-commute mult-right-isotone star-isotone* **by** *auto*
also have ... $\leq ?c$
using *comp-inf.mult-right-isotone comp-isotone conv-isotone inf.commute mult-right-isotone pp-isotone star-isotone sup.cobounded1* **by** *presburger*
finally have *2*: $v^T \leq ?c \wedge v \leq ?c^T$
by (*metis conv-isotone conv-involutive*)
have $t \leq v * v^T$
using *assms*(*2*) **by** *auto*
hence *3*: $t \leq ?c^T * ?c$
using *2 order-trans mult-isotone* **by** *blast*
have $e \leq v * top \sqcap --g$
by (*metis assms*(*4*,*5*) *inf.bounded-iff inf.sup-left-divisibility mult-right-isotone top.extremum*)
hence $e \leq v * top \sqcap top * e \sqcap --g$
by (*simp add*: *top-left-mult-increasing inf.boundedI*)
hence $e \leq v * top * e \sqcap --g$
by (*metis comp-inf-covector inf.absorb2 mult-assoc top.extremum*)
hence $t \sqcup e \leq (v * v^T \sqcap --g) \sqcup (v * top * e \sqcap --g)$
using *assms*(*2*) *sup-mono* **by** *blast*
also have ... $= v * ?v^T \sqcap --g$
by (*simp add*: *inf-sup-distrib2 mult-assoc mult-left-dist-sup conv-dist-comp conv-dist-sup*)
also have ... $\leq --?G$
using *assms*(*6*) *comp-left-increasing-sup inf.sup-left-isotone pp-dist-inf* **by** *auto*
finally have *4*: $t \sqcup e \leq --?G$
.
have $e \leq e * e^T * e$
by (*simp add*: *ex231c*)
also have ... $\leq v * -v^T * -v * v^T * e$
by (*metis assms*(*4*) *mult-left-isotone conv-isotone conv-dist-comp mult-assoc mult-isotone conv-involutive conv-complement inf.boundedE*)
also have ... $\leq v * top * v^T * e$
by (*metis mult-assoc mult-left-isotone mult-right-isotone top.extremum*)
also have ... $= v * r^T * t^\star * e$
using *assms*(*3*,*5*) **by** (*simp add*: *mult-assoc*)
also have ... $\leq v * r^T * (t \sqcup e)^\star$
by (*simp add*: *comp-associative mult-right-isotone star.circ-mult-upper-bound star.circ-sub-dist-1 star-isotone sup-commute*)

also have ... $\leq v * ?c$
using *4* **by** (*simp add*: *mult-assoc mult-right-isotone star-isotone*)
also have ... $\leq ?c^T * ?c$
using *2* **by** (*simp add*: *mult-left-isotone*)
finally show *?thesis*
using *3* **by** *simp*
qed

lemma *span-inv*:
assumes $e \leq v * -v^T$
and *vector v*
and *arc e*
and $t \leq (v * v^T) \sqcap g$
and $g^T = g$
and $v^T = r^T * t^\star$
and *injective r*
and $r^T \leq v^T$
and $r^T * ((v * v^T) \sqcap g)^\star \leq r^T * t^\star$
shows $r^T * (((v \sqcup e^T * top) * (v \sqcup e^T * top)^T) \sqcap g)^\star \leq r^T * (t \sqcup e)^\star$
proof $-$
let $?d = (v * v^T) \sqcap g$
have *1*: $(v \sqcup e^T * top) * (v \sqcup e^T * top)^T = v * v^T \sqcup v * v^T * e \sqcup e^T * v * v^T \sqcup e^T * e$
using *assms*(*1*$-$*3*) *ve-dist* **by** *simp*
have $t^T \leq ?d^T$
using *assms*(*4*) *conv-isotone* **by** *simp*
also have ... $= (v * v^T) \sqcap g^T$
by (*simp add*: *conv-dist-comp conv-dist-inf*)
also have ... $= ?d$
by (*simp add*: *assms*(*5*))
finally have *2*: $t^T \leq ?d$
.
have $v * v^T = (r^T * t^\star)^T * (r^T * t^\star)$
by (*metis assms*(*6*) *conv-involutive*)
also have ... $= t^{T\star} * (r * r^T) * t^\star$
by (*simp add*: *comp-associative conv-dist-comp conv-star-commute*)
also have ... $\leq t^{T\star} * 1 * t^\star$
by (*simp add*: *assms*(*7*) *mult-left-isotone star-right-induct-mult-iff star-sub-one*)
also have ... $= t^{T\star} * t^\star$
by *simp*
also have ... $\leq ?d^\star * t^\star$
using *2* **by** (*simp add*: *comp-left-isotone star.circ-isotone*)
also have ... $\leq ?d^\star * ?d^\star$
using *assms*(*4*) *mult-right-isotone star-isotone* **by** *simp*
also have *3*: ... $= ?d^\star$
by (*simp add*: *star.circ-transitive-equal*)
finally have *4*: $v * v^T \leq ?d^\star$
.

have *5*: $r^T * ?d^\star * (v * v^T \sqcap g) \le r^T * ?d^\star$
by (*simp add*: *comp-associative mult-right-isotone star.circ-plus-same star.left-plus-below-circ*)
have $r^T * ?d^\star * (v * v^T * e \sqcap g) \le r^T * ?d^\star * v * v^T * e$
by (*simp add*: *comp-associative comp-right-isotone*)
also have ... $\le r^T * ?d^\star * e$
using *3 4* **by** (*metis comp-associative comp-isotone eq-refl*)
finally have *6*: $r^T * ?d^\star * (v * v^T * e \sqcap g) \le r^T * ?d^\star * e$
.
have *7*: $\forall x . r^T * (1 \sqcup v * v^T) * e^T * x = bot$
proof
fix *x*
have $r^T * (1 \sqcup v * v^T) * e^T * x \le r^T * (1 \sqcup v * v^T) * e^T * top$
by (*simp add*: *mult-right-isotone*)
also have ... $= r^T * e^T * top \sqcup r^T * v * v^T * e^T * top$
by (*simp add*: *comp-associative mult-left-dist-sup mult-right-dist-sup*)
also have ... $= r^T * e^T * top$
by (*metis assms*(*1*,*2*) *mult-assoc mult-right-dist-sup mult-right-zero sup-bot-right vTeT*)
also have ... $\le v^T * e^T * top$
by (*simp add*: *assms*(*8*) *comp-isotone*)
also have ... $= bot$
using *vTeT assms*(*1*,*2*) **by** *simp*
finally show $r^T * (1 \sqcup v * v^T) * e^T * x = bot$
by (*simp add*: *le-bot*)
qed
have $r^T * ?d^\star * (e^T * v * v^T \sqcap g) \le r^T * ?d^\star * e^T * v * v^T$
by (*simp add*: *comp-associative comp-right-isotone*)
also have ... $\le r^T * (1 \sqcup v * v^T) * e^T * v * v^T$
by (*metis assms*(*2*) *star.circ-isotone vector-vector-star inf-le1 comp-associative comp-right-isotone comp-left-isotone*)
also have ... $= bot$
using *7* **by** *simp*
finally have *8*: $r^T * ?d^\star * (e^T * v * v^T \sqcap g) = bot$
by (*simp add*: *le-bot*)
have $r^T * ?d^\star * (e^T * e \sqcap g) \le r^T * ?d^\star * e^T * e$
by (*simp add*: *comp-associative comp-right-isotone*)
also have ... $\le r^T * (1 \sqcup v * v^T) * e^T * e$
by (*metis assms*(*2*) *star.circ-isotone vector-vector-star inf-le1 comp-associative comp-right-isotone comp-left-isotone*)
also have ... $= bot$
using *7* **by** *simp*
finally have *9*: $r^T * ?d^\star * (e^T * e \sqcap g) = bot$
by (*simp add*: *le-bot*)
have $r^T * ?d^\star * ((v \sqcup e^T * top) * (v \sqcup e^T * top)^T \sqcap g) = r^T * ?d^\star * ((v * v^T \sqcup v * v^T * e \sqcup e^T * v * v^T \sqcup e^T * e) \sqcap g)$
using *1* **by** *simp*
also have ... $= r^T * ?d^\star * ((v * v^T \sqcap g) \sqcup (v * v^T * e \sqcap g) \sqcup (e^T * v * v^T \sqcap g) \sqcup (e^T * e \sqcap g))$

by (*simp add*: *inf-sup-distrib2*)
also have ... $= r^T * ?d^\star * (v * v^T \sqcap g) \sqcup r^T * ?d^\star * (v * v^T * e \sqcap g) \sqcup r^T * ?d^\star * (e^T * v * v^T \sqcap g) \sqcup r^T * ?d^\star * (e^T * e \sqcap g)$
by (*simp add*: *comp-left-dist-sup*)
also have ... $= r^T * ?d^\star * (v * v^T \sqcap g) \sqcup r^T * ?d^\star * (v * v^T * e \sqcap g)$
using *8 9* **by** *simp*
also have ... $\leq r^T * ?d^\star \sqcup r^T * ?d^\star * e$
using *5 6 sup.mono* **by** *simp*
also have ... $= r^T * ?d^\star * (1 \sqcup e)$
by (*simp add*: *mult-left-dist-sup*)
finally have *10*: $r^T * ?d^\star * ((v \sqcup e^T * top) * (v \sqcup e^T * top)^T \sqcap g) \leq r^T * ?d^\star * (1 \sqcup e)$
by *simp*
have $r^T * ?d^\star * e * (v * v^T \sqcap g) \leq r^T * ?d^\star * e * v * v^T$
by (*simp add*: *comp-associative comp-right-isotone*)
also have ... $= bot$
by (*metis assms*(*1*,*2*) *comp-associative comp-right-zero ev comp-left-zero*)
finally have *11*: $r^T * ?d^\star * e * (v * v^T \sqcap g) = bot$
by (*simp add*: *le-bot*)
have $r^T * ?d^\star * e * (v * v^T * e \sqcap g) \leq r^T * ?d^\star * e * v * v^T * e$
by (*simp add*: *comp-associative comp-right-isotone*)
also have ... $= bot$
by (*metis assms*(*1*,*2*) *comp-associative comp-right-zero ev comp-left-zero*)
finally have *12*: $r^T * ?d^\star * e * (v * v^T * e \sqcap g) = bot$
by (*simp add*: *le-bot*)
have $r^T * ?d^\star * e * (e^T * v * v^T \sqcap g) \leq r^T * ?d^\star * e * e^T * v * v^T$
by (*simp add*: *comp-associative comp-right-isotone*)
also have ... $\leq r^T * ?d^\star * 1 * v * v^T$
by (*metis assms*(*3*) *arc-injective comp-associative comp-left-isotone comp-right-isotone*)
also have ... $= r^T * ?d^\star * v * v^T$
by *simp*
also have ... $\leq r^T * ?d^\star * ?d^\star$
using *4* **by** (*simp add*: *mult-right-isotone mult-assoc*)
also have ... $= r^T * ?d^\star$
by (*simp add*: *star.circ-transitive-equal comp-associative*)
finally have *13*: $r^T * ?d^\star * e * (e^T * v * v^T \sqcap g) \leq r^T * ?d^\star$
.
have $r^T * ?d^\star * e * (e^T * e \sqcap g) \leq r^T * ?d^\star * e * e^T * e$
by (*simp add*: *comp-associative comp-right-isotone*)
also have ... $\leq r^T * ?d^\star * 1 * e$
by (*metis assms*(*3*) *arc-injective comp-associative comp-left-isotone comp-right-isotone*)
also have ... $= r^T * ?d^\star * e$
by *simp*
finally have *14*: $r^T * ?d^\star * e * (e^T * e \sqcap g) \leq r^T * ?d^\star * e$
.
have $r^T * ?d^\star * e * ((v \sqcup e^T * top) * (v \sqcup e^T * top)^T \sqcap g) = r^T * ?d^\star * e * ((v * v^T \sqcup v * v^T * e \sqcup e^T * v * v^T \sqcup e^T * e) \sqcap g)$

using *1* **by** *simp*
also have ... $= r^T * ?d^\star * e * ((v * v^T \sqcap g) \sqcup (v * v^T * e \sqcap g) \sqcup (e^T * v * v^T \sqcap g) \sqcup (e^T * e \sqcap g))$
by (*simp add*: *inf-sup-distrib2*)
also have ... $= r^T * ?d^\star * e * (v * v^T \sqcap g) \sqcup r^T * ?d^\star * e * (v * v^T * e \sqcap g) \sqcup r^T * ?d^\star * e * (e^T * v * v^T \sqcap g) \sqcup r^T * ?d^\star * e * (e^T * e \sqcap g)$
by (*simp add*: *comp-left-dist-sup*)
also have ... $= r^T * ?d^\star * e * (e^T * v * v^T \sqcap g) \sqcup r^T * ?d^\star * e * (e^T * e \sqcap g)$
using *11 12* **by** *simp*
also have ... $\leq r^T * ?d^\star \sqcup r^T * ?d^\star * e$
using *13 14 sup-mono* **by** *simp*
also have ... $= r^T * ?d^\star * (1 \sqcup e)$
by (*simp add*: *mult-left-dist-sup*)
finally have *15*: $r^T * ?d^\star * e * ((v \sqcup e^T * top) * (v \sqcup e^T * top)^T \sqcap g) \leq r^T * ?d^\star * (1 \sqcup e)$
by *simp*
have $r^T \leq r^T * ?d^\star$
using *mult-right-isotone star.circ-reflexive* **by** *fastforce*
also have ... $\leq r^T * ?d^\star * (1 \sqcup e)$
by (*simp add*: *semiring.distrib-left*)
finally have *16*: $r^T \leq r^T * ?d^\star * (1 \sqcup e)$
.
have $r^T * ?d^\star * (1 \sqcup e) * ((v \sqcup e^T * top) * (v \sqcup e^T * top)^T \sqcap g) = r^T * ?d^\star * ((v \sqcup e^T * top) * (v \sqcup e^T * top)^T \sqcap g) \sqcup r^T * ?d^\star * e * ((v \sqcup e^T * top) * (v \sqcup e^T * top)^T \sqcap g)$
by (*simp add*: *semiring.distrib-left semiring.distrib-right*)
also have ... $\leq r^T * ?d^\star * (1 \sqcup e)$
using *10 15 le-supI* **by** *simp*
finally have $r^T * ?d^\star * (1 \sqcup e) * ((v \sqcup e^T * top) * (v \sqcup e^T * top)^T \sqcap g) \leq r^T * ?d^\star * (1 \sqcup e)$
.
hence $r^T \sqcup r^T * ?d^\star * (1 \sqcup e) * ((v \sqcup e^T * top) * (v \sqcup e^T * top)^T \sqcap g) \leq r^T * ?d^\star * (1 \sqcup e)$
using *16 sup-least* **by** *simp*
hence $r^T * ((v \sqcup e^T * top) * (v \sqcup e^T * top)^T \sqcap g)^\star \leq r^T * ?d^\star * (1 \sqcup e)$
by (*simp add*: *star-right-induct*)
also have ... $\leq r^T * t^\star * (1 \sqcup e)$
by (*simp add*: *assms*(*9*) *mult-left-isotone*)
also have ... $\leq r^T * (t \sqcup e)^\star$
by (*simp add*: *star-one-sup-below*)
finally show *?thesis*
.
qed

4.1.2 Exchange gives Spanning Trees

The following abbreviations are used in the spanning tree application using Prim's algorithm to construct the new tree for the exchange property. It is obtained by replacing an edge with one that has minimal weight and reversing the path connecting these edges. Here, w represents a weighted graph, v represents a set of nodes and e represents an edge.

abbreviation *prim-E* :: $'a \Rightarrow 'a \Rightarrow 'a \Rightarrow 'a$ **where** *prim-E w v e* $\equiv w \sqcap --v * -v^T \sqcap top * e * w^{T\star}$
abbreviation *prim-P* :: $'a \Rightarrow 'a \Rightarrow 'a \Rightarrow 'a$ **where** *prim-P w v e* $\equiv w \sqcap -v * -v^T \sqcap top * e * w^{T\star}$
abbreviation *prim-EP* :: $'a \Rightarrow 'a \Rightarrow 'a \Rightarrow 'a$ **where** *prim-EP w v e* $\equiv w \sqcap -v^T \sqcap top * e * w^{T\star}$
abbreviation *prim-W* :: $'a \Rightarrow 'a \Rightarrow 'a \Rightarrow 'a$ **where** *prim-W w v e* $\equiv (w \sqcap -(prim\text{-}EP\ w\ v\ e)) \sqcup (prim\text{-}P\ w\ v\ e)^T \sqcup e$

The lemmas in this section are used to show that the relation after exchange represents a spanning tree. The results in the next section are used to show that it is a minimum spanning tree.

lemma *exchange-injective-3*:
  assumes $e \leq v * -v^T$
    and *vector v*
   shows $(w \sqcap -(prim\text{-}EP\ w\ v\ e)) * e^T = bot$
proof –
  have *1*: $top * e \leq -v^T$
   by (*simp add*: *assms schroeder-4-p vTeT*)
  have $top * e \leq top * e * w^{T\star}$
   using *sup-right-divisibility star.circ-back-loop-fixpoint* **by** *blast*
  hence $top * e \leq -v^T \sqcap top * e * w^{T\star}$
   using *1* **by** *simp*
  hence $top * e \leq -(w \sqcap -prim\text{-}EP\ w\ v\ e)$
   by (*metis inf.assoc inf-import-p le-infI2 p-antitone p-antitone-iff*)
  hence $(w \sqcap -(prim\text{-}EP\ w\ v\ e)) * e^T \leq bot$
   using *p-top schroeder-4-p* **by** *blast*
  thus *?thesis*
   using *le-bot* **by** *simp*
qed

lemma *exchange-injective-6*:
  assumes *arc e*
    and *forest w*
   shows $(prim\text{-}P\ w\ v\ e)^T * e^T = bot$
proof –
  have $e^T * top * e \leq --1$
   by (*simp add*: *assms(1) p-antitone p-antitone-iff point-injective*)
  hence *1*: $e * -1 * e^T \leq bot$
   by (*metis conv-involutive p-top triple-schroeder-p*)
  have $(prim\text{-}P\ w\ v\ e)^T * e^T \leq (w \sqcap top * e * w^{T\star})^T * e^T$
   using *comp-inf.mult-left-isotone conv-dist-inf mult-left-isotone* **by** *simp*
  also have ... $= (w^T \sqcap w^{T\star T} * e^T * top) * e^T$
   by (*simp add*: *comp-associative conv-dist-comp conv-dist-inf*)
  also have ... $= w^\star * e^T * top \sqcap w^T * e^T$
   by (*simp add*: *conv-star-commute inf-vector-comp*)
  also have ... $\leq (w^T \sqcap w^\star * e^T * top * e) * (e^T \sqcap w^+ * e^T * top)$

by (*metis dedekind mult-assoc conv-involutive inf-commute*)
also have ... $\leq (w^\star * e^T * top * e) * (w^+ * e^T * top)$
by (*simp add*: *mult-isotone*)
also have ... $\leq (top * e) * (w^+ * e^T * top)$
by (*simp add*: *mult-left-isotone*)
also have ... $= top * e * w^+ * e^T * top$
using *mult-assoc* **by** *simp*
also have ... $\leq top * e * -1 * e^T * top$
using *assms*(*2*) *mult-left-isotone mult-right-isotone* **by** *simp*
also have ... $\leq bot$
using *1* **by** (*metis le-bot semiring.mult-not-zero mult-assoc*)
finally show *?thesis*
using *le-bot* **by** *simp*
qed

The graph after exchanging is injective.

lemma *exchange-injective*:
assumes *arc e*
and $e \leq v * -v^T$
and *forest w*
and *vector v*
shows *injective* (*prim-W w v e*)
proof –
have *1*: $(w \sqcap -(prim\text{-}EP\ w\ v\ e)) * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^T \leq 1$
proof –
have $(w \sqcap -(prim\text{-}EP\ w\ v\ e)) * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^T \leq w * w^T$
by (*simp add*: *comp-isotone conv-isotone*)
also have ... ≤ 1
by (*simp add*: *assms*(*3*))
finally show *?thesis*
.
qed
have *2*: $(w \sqcap -(prim\text{-}EP\ w\ v\ e)) * (prim\text{-}P\ w\ v\ e)^{TT} \leq 1$
proof –
have $top * (prim\text{-}P\ w\ v\ e)^T = top * (w^T \sqcap -v * -v^T \sqcap w^{T\star T} * e^T * top)$
by (*simp add*: *comp-associative conv-complement conv-dist-comp conv-dist-inf*)
also have ... $= top * e * w^{T\star} * (w^T \sqcap -v * -v^T)$
by (*metis comp-inf-vector conv-dist-comp conv-involutive inf-top-left mult-assoc*)
also have ... $\leq top * e * w^{T\star} * (w^T \sqcap top * -v^T)$
using *comp-inf.mult-right-isotone mult-left-isotone mult-right-isotone* **by** *simp*
also have ... $= top * e * w^{T\star} * w^T \sqcap -v^T$
by (*metis assms*(*4*) *comp-inf-covector vector-conv-compl*)
also have ... $\leq -v^T \sqcap top * e * w^{T\star}$
by (*simp add*: *comp-associative comp-isotone inf.coboundedI1 star.circ-plus-same star.left-plus-below-circ*)
finally have $top * (prim\text{-}P\ w\ v\ e)^T \leq -(w \sqcap -prim\text{-}EP\ w\ v\ e)$

by (*metis inf.assoc inf-import-p le-infI2 p-antitone p-antitone-iff*)
hence $(w \sqcap -(\mathit{prim\text{-}EP}\ w\ v\ e)) * (\mathit{prim\text{-}P}\ w\ v\ e)^{TT} \leq \mathit{bot}$
using *p-top schroeder-4-p* **by** *blast*
thus *?thesis*
by (*simp add*: *bot-unique*)
qed
have *3*: $(w \sqcap -(\mathit{prim\text{-}EP}\ w\ v\ e)) * e^T \leq 1$
by (*metis assms*(*2*,*4*) *exchange-injective-3 bot-least*)
have *4*: $(\mathit{prim\text{-}P}\ w\ v\ e)^T * (w \sqcap -(\mathit{prim\text{-}EP}\ w\ v\ e))^T \leq 1$
using *2 conv-dist-comp coreflexive-symmetric* **by** *fastforce*
have *5*: $(\mathit{prim\text{-}P}\ w\ v\ e)^T * (\mathit{prim\text{-}P}\ w\ v\ e)^{TT} \leq 1$
proof −
have $(\mathit{prim\text{-}P}\ w\ v\ e)^T * (\mathit{prim\text{-}P}\ w\ v\ e)^{TT} \leq (\mathit{top} * e * w^{T\star})^T * (\mathit{top} * e * w^{T\star})$
by (*simp add*: *conv-dist-inf mult-isotone*)
also have ... $= w^\star * e^T * \mathit{top} * \mathit{top} * e * w^{T\star}$
using *conv-star-commute conv-dist-comp conv-involutive conv-top mult-assoc* **by** *presburger*
also have ... $= w^\star * e^T * \mathit{top} * e * w^{T\star}$
by (*simp add*: *comp-associative*)
also have ... $\leq w^\star * 1 * w^{T\star}$
by (*metis comp-left-isotone comp-right-isotone mult-assoc assms*(*1*) *point-injective*)
finally have $(\mathit{prim\text{-}P}\ w\ v\ e)^T * (\mathit{prim\text{-}P}\ w\ v\ e)^{TT} \leq w^\star * w^{T\star} \sqcap w^T * w$
by (*simp add*: *conv-isotone inf.left-commute inf.sup-monoid.add-commute mult-isotone*)
also have ... ≤ 1
by (*simp add*: *assms*(*3*) *forest-separate*)
finally show *?thesis*
.
qed
have *6*: $(\mathit{prim\text{-}P}\ w\ v\ e)^T * e^T \leq 1$
using *assms exchange-injective-6 bot-least* **by** *simp*
have *7*: $e * (w \sqcap -(\mathit{prim\text{-}EP}\ w\ v\ e))^T \leq 1$
using *3* **by** (*metis conv-dist-comp conv-involutive coreflexive-symmetric*)
have *8*: $e * (\mathit{prim\text{-}P}\ w\ v\ e)^{TT} \leq 1$
using *6 conv-dist-comp coreflexive-symmetric* **by** *fastforce*
have *9*: $e * e^T \leq 1$
by (*simp add*: *assms*(*1*) *arc-injective*)
have $(\mathit{prim\text{-}W}\ w\ v\ e) * (\mathit{prim\text{-}W}\ w\ v\ e)^T = (w \sqcap -(\mathit{prim\text{-}EP}\ w\ v\ e)) * (w \sqcap -(\mathit{prim\text{-}EP}\ w\ v\ e))^T \sqcup (w \sqcap -(\mathit{prim\text{-}EP}\ w\ v\ e)) * (\mathit{prim\text{-}P}\ w\ v\ e)^{TT} \sqcup (w \sqcap -(\mathit{prim\text{-}EP}\ w\ v\ e)) * e^T \sqcup (\mathit{prim\text{-}P}\ w\ v\ e)^T * (w \sqcap -(\mathit{prim\text{-}EP}\ w\ v\ e))^T \sqcup (\mathit{prim\text{-}P}\ w\ v\ e)^T * (\mathit{prim\text{-}P}\ w\ v\ e)^{TT} \sqcup (\mathit{prim\text{-}P}\ w\ v\ e)^T * e^T \sqcup e * (w \sqcap -(\mathit{prim\text{-}EP}\ w\ v\ e))^T \sqcup e * (\mathit{prim\text{-}P}\ w\ v\ e)^{TT} \sqcup e * e^T$
using *comp-left-dist-sup comp-right-dist-sup conv-dist-sup sup.assoc* **by** *simp*
also have ... ≤ 1
using *1 2 3 4 5 6 7 8 9* **by** *simp*
finally show *?thesis*
.

qed

lemma *pv*:
  assumes *vector v*
    shows $(prim\text{-}P\ w\ v\ e)^T * v = bot$
proof −
  have $(prim\text{-}P\ w\ v\ e)^T * v \le (-v * -v^T)^T * v$
    by (*meson conv-isotone inf-le1 inf-le2 mult-left-isotone order-trans*)
  also have ... $= -v * -v^T * v$
    by (*simp add*: *conv-complement conv-dist-comp*)
  also have ... $= bot$
    by (*simp add*: *assms covector-vector-comp mult-assoc*)
  finally show *?thesis*
    by (*simp add*: *order.antisym*)
qed

lemma *vector-pred-inv*:
  assumes *arc e*
      and $e \le v * -v^T$
      and *forest w*
      and *vector v*
      and $w * v \le v$
    shows $(prim\text{-}W\ w\ v\ e) * (v \sqcup e^T * top) \le v \sqcup e^T * top$
proof −
  have $(prim\text{-}W\ w\ v\ e) * e^T * top = (w \sqcap -(prim\text{-}EP\ w\ v\ e)) * e^T * top \sqcup (prim\text{-}P\ w\ v\ e)^T * e^T * top \sqcup e * e^T * top$
    by (*simp add*: *mult-right-dist-sup*)
  also have ... $= e * e^T * top$
   using *assms exchange-injective-3 exchange-injective-6 comp-left-zero* **by** *simp*
  also have ... $\le v * -v^T * e^T * top$
    by (*simp add*: *assms*(*2*) *comp-isotone*)
  also have ... $\le v * top$
    by (*simp add*: *comp-associative mult-right-isotone*)
  also have ... $= v$
    by (*simp add*: *assms*(*4*))
  finally have *1*: $(prim\text{-}W\ w\ v\ e) * e^T * top \le v$
    .
  have $(prim\text{-}W\ w\ v\ e) * v = (w \sqcap -(prim\text{-}EP\ w\ v\ e)) * v \sqcup (prim\text{-}P\ w\ v\ e)^T * v \sqcup e * v$
    by (*simp add*: *mult-right-dist-sup*)
  also have ... $= (w \sqcap -(prim\text{-}EP\ w\ v\ e)) * v$
    by (*metis assms*(*2*,*4*) *pv ev sup-bot-right*)
  also have ... $\le w * v$
    by (*simp add*: *mult-left-isotone*)
  finally have *2*: $(prim\text{-}W\ w\ v\ e) * v \le v$
    using *assms*(*5*) *order-trans* **by** *blast*
  have $(prim\text{-}W\ w\ v\ e) * (v \sqcup e^T * top) = (prim\text{-}W\ w\ v\ e) * v \sqcup (prim\text{-}W\ w\ v\ e) * e^T * top$
    by (*simp add*: *semiring.distrib-left mult-assoc*)

also have ... $\leq v$
using *1 2* **by** *simp*
also have ... $\leq v \sqcup e^T * top$
by *simp*
finally show *?thesis*
.
qed

The graph after exchanging is acyclic.

lemma *exchange-acyclic*:
assumes *vector v*
and $e \leq v * -v^T$
and $w * v \leq v$
and *acyclic w*
shows *acyclic* (*prim-W w v e*)
proof −
have *1*: $(prim\text{-}P\ w\ v\ e)^T * e = bot$
proof −
have $(prim\text{-}P\ w\ v\ e)^T * e \leq (-v * -v^T)^T * e$
by (*meson conv-order dual-order.trans inf.cobounded1 inf.cobounded2 mult-left-isotone*)
also have ... $= -v * -v^T * e$
by (*simp add*: *conv-complement conv-dist-comp*)
also have ... $\leq -v * -v^T * v * -v^T$
by (*simp add*: *assms*(*2*) *comp-associative mult-right-isotone*)
also have ... $= bot$
by (*simp add*: *assms*(*1*) *covector-vector-comp mult-assoc*)
finally show *?thesis*
by (*simp add*: *bot-unique*)
qed
have *2*: $e * e = bot$
using *assms*(*1*,*2*) *ee* **by** *auto*
have *3*: $(w \sqcap -(prim\text{-}EP\ w\ v\ e)) * (prim\text{-}P\ w\ v\ e)^T = bot$
proof −
have $top * (prim\text{-}P\ w\ v\ e) \leq top * (-v * -v^T \sqcap top * e * w^{T\star})$
using *comp-inf.mult-semi-associative mult-right-isotone* **by** *auto*
also have ... $\leq top * -v * -v^T \sqcap top * top * e * w^{T\star}$
by (*simp add*: *comp-inf-covector mult-assoc*)
also have ... $\leq top * -v^T \sqcap top * e * w^{T\star}$
using *mult-left-isotone top.extremum inf-mono* **by** *presburger*
also have ... $= -v^T \sqcap top * e * w^{T\star}$
by (*simp add*: *assms*(*1*) *vector-conv-compl*)
finally have $top * (prim\text{-}P\ w\ v\ e) \leq -(w \sqcap -prim\text{-}EP\ w\ v\ e)$
by (*metis inf.assoc inf-import-p le-infI2 p-antitone p-antitone-iff*)
hence $(w \sqcap -(prim\text{-}EP\ w\ v\ e)) * (prim\text{-}P\ w\ v\ e)^T \leq bot$
using *p-top schroeder-4-p* **by** *blast*
thus *?thesis*
using *bot-unique* **by** *blast*
qed

hence *4*: $(w \sqcap -(prim\text{-}EP\ w\ v\ e)) * (prim\text{-}P\ w\ v\ e)^{T\star} = w \sqcap -(prim\text{-}EP\ w\ v\ e)$
using *star-absorb* **by** *blast*
hence *5*: $(w \sqcap -(prim\text{-}EP\ w\ v\ e))^{+} * (prim\text{-}P\ w\ v\ e)^{T\star} = (w \sqcap -(prim\text{-}EP\ w\ v\ e))^{+}$
by (*metis star-plus mult-assoc*)
hence *6*: $(w \sqcap -(prim\text{-}EP\ w\ v\ e))^{\star} * (prim\text{-}P\ w\ v\ e)^{T\star} = (w \sqcap -(prim\text{-}EP\ w\ v\ e))^{+} \sqcup (prim\text{-}P\ w\ v\ e)^{T\star}$
by (*metis star.circ-loop-fixpoint mult-assoc*)
have *7*: $(w \sqcap -(prim\text{-}EP\ w\ v\ e))^{+} * e \leq v * top$
proof −
have $e \leq v * top$
using *assms*(*2*) *dual-order.trans mult-right-isotone top-greatest* **by** *blast*
hence *8*: $e \sqcup w * v * top \leq v * top$
by (*simp add*: *assms*(*1*,*3*) *comp-associative*)
have $(w \sqcap -(prim\text{-}EP\ w\ v\ e))^{+} * e \leq w^{+} * e$
by (*simp add*: *comp-isotone star-isotone*)
also have ... $\leq w^{\star} * e$
by (*simp add*: *mult-left-isotone star.left-plus-below-circ*)
also have ... $\leq v * top$
using *8* **by** (*simp add*: *comp-associative star-left-induct*)
finally show *?thesis*
.
qed
have *9*: $(prim\text{-}P\ w\ v\ e)^{T} * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^{+} * e = bot$
proof −
have $(prim\text{-}P\ w\ v\ e)^{T} * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^{+} * e \leq (prim\text{-}P\ w\ v\ e)^{T} * v * top$
using *7* **by** (*simp add*: *mult-assoc mult-right-isotone*)
also have ... $= bot$
by (*simp add*: *assms*(*1*) *pv*)
finally show *?thesis*
using *bot-unique* **by** *blast*
qed
have *10*: $e * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^{+} * e = bot$
proof −
have $e * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^{+} * e \leq e * v * top$
using *7* **by** (*simp add*: *mult-assoc mult-right-isotone*)
also have ... $\leq v * -v^{T} * v * top$
by (*simp add*: *assms*(*2*) *mult-left-isotone*)
also have ... $= bot$
by (*simp add*: *assms*(*1*) *covector-vector-comp mult-assoc*)
finally show *?thesis*
using *bot-unique* **by** *blast*
qed
have *11*: $e * (prim\text{-}P\ w\ v\ e)^{T\star} * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^{\star} \leq v * -v^{T}$
proof −
have *12*: $-v^{T} * w \leq -v^{T}$
by (*metis assms*(*3*) *conv-complement order-lesseq-imp pp-increasing schroeder-6-p*)

have $v * -v^T * (w \sqcap -(\textit{prim-EP}\ w\ v\ e)) \leq v * -v^T * w$
by (*simp add*: *comp-isotone star-isotone*)
also have ... $\leq v * -v^T$
using *12* **by** (*simp add*: *comp-isotone comp-associative*)
finally have *13*: $v * -v^T * (w \sqcap -(\textit{prim-EP}\ w\ v\ e)) \leq v * -v^T$
.
have *14*: $(\textit{prim-P}\ w\ v\ e)^T \leq -v * -v^T$
by (*metis conv-complement conv-dist-comp conv-involutive conv-order inf-le1 inf-le2 order-trans*)
have $e * (\textit{prim-P}\ w\ v\ e)^{T\star} \leq v * -v^T * (\textit{prim-P}\ w\ v\ e)^{T\star}$
by (*simp add*: *assms*(*2*) *mult-left-isotone*)
also have ... $= v * -v^T \sqcup v * -v^T * (\textit{prim-P}\ w\ v\ e)^{T+}$
by (*metis mult-assoc star.circ-back-loop-fixpoint star-plus sup-commute*)
also have ... $= v * -v^T \sqcup v * -v^T * (\textit{prim-P}\ w\ v\ e)^{T\star} * (\textit{prim-P}\ w\ v\ e)^T$
by (*simp add*: *mult-assoc star-plus*)
also have ... $\leq v * -v^T \sqcup v * -v^T * (\textit{prim-P}\ w\ v\ e)^{T\star} * -v * -v^T$
using *14* *mult-assoc mult-right-isotone sup-right-isotone* **by** *simp*
also have ... $\leq v * -v^T \sqcup v * \textit{top} * -v^T$
by (*metis top-greatest mult-right-isotone mult-left-isotone mult-assoc sup-right-isotone*)
also have ... $= v * -v^T$
by (*simp add*: *assms*(*1*))
finally have $e * (\textit{prim-P}\ w\ v\ e)^{T\star} * (w \sqcap -(\textit{prim-EP}\ w\ v\ e))^\star \leq v * -v^T * (w \sqcap -(\textit{prim-EP}\ w\ v\ e))^\star$
by (*simp add*: *mult-left-isotone*)
also have ... $\leq v * -v^T$
using *13* **by** (*simp add*: *star-right-induct-mult*)
finally show *?thesis*
.
qed
have *15*: $(w \sqcap -(\textit{prim-EP}\ w\ v\ e))^+ * (\textit{prim-P}\ w\ v\ e)^{T\star} * (w \sqcap -(\textit{prim-EP}\ w\ v\ e))^\star \leq -1$
proof −
have $(w \sqcap -(\textit{prim-EP}\ w\ v\ e))^+ * (\textit{prim-P}\ w\ v\ e)^{T\star} * (w \sqcap -(\textit{prim-EP}\ w\ v\ e))^\star = (w \sqcap -(\textit{prim-EP}\ w\ v\ e))^+ * (w \sqcap -(\textit{prim-EP}\ w\ v\ e))^\star$
using *5* **by** *simp*
also have ... $= (w \sqcap -(\textit{prim-EP}\ w\ v\ e))^+$
by (*simp add*: *mult-assoc star.circ-transitive-equal*)
also have ... $\leq w^+$
by (*simp add*: *comp-isotone star-isotone*)
finally show *?thesis*
using *assms*(*4*) **by** *simp*
qed
have *16*: $(\textit{prim-P}\ w\ v\ e)^T * (w \sqcap -(\textit{prim-EP}\ w\ v\ e))^\star * (\textit{prim-P}\ w\ v\ e)^{T\star} * (w \sqcap -(\textit{prim-EP}\ w\ v\ e))^\star \leq -1$
proof −
have $(w \sqcap -(\textit{prim-EP}\ w\ v\ e))^+ * (\textit{prim-P}\ w\ v\ e)^{T+} \leq (w \sqcap -(\textit{prim-EP}\ w\ v\ e))^+ * (\textit{prim-P}\ w\ v\ e)^{T\star}$
by (*simp add*: *mult-right-isotone star.left-plus-below-circ*)

also have ... $= (w \sqcap -(\textit{prim-EP } w\ v\ e))^+$
using *5* **by** *simp*
also have ... $\leq w^+$
by (*simp add*: *comp-isotone star-isotone*)
finally have $(w \sqcap -(\textit{prim-EP } w\ v\ e))^+ * (\textit{prim-P } w\ v\ e)^{T+} \leq -1$
using *assms*(*4*) **by** *simp*
hence *17*: $(\textit{prim-P } w\ v\ e)^{T+} * (w \sqcap -(\textit{prim-EP } w\ v\ e))^+ \leq -1$
by (*simp add*: *comp-commute-below-diversity*)
have $(\textit{prim-P } w\ v\ e)^{T+} \leq w^{T+}$
by (*simp add*: *comp-isotone conv-dist-inf inf.left-commute inf.sup-monoid.add-commute star-isotone*)
also have ... $= w^{+T}$
by (*simp add*: *conv-dist-comp conv-star-commute star-plus*)
also have ... ≤ -1
using *assms*(*4*) *conv-complement conv-isotone* **by** *force*
finally have *18*: $(\textit{prim-P } w\ v\ e)^{T+} \leq -1$
.
have $(\textit{prim-P } w\ v\ e)^T * (w \sqcap -(\textit{prim-EP } w\ v\ e))^\star * (\textit{prim-P } w\ v\ e)^{T\star} * (w \sqcap -(\textit{prim-EP } w\ v\ e))^\star = (\textit{prim-P } w\ v\ e)^T * ((w \sqcap -(\textit{prim-EP } w\ v\ e))^+ \sqcup (\textit{prim-P } w\ v\ e)^{T\star}) * (w \sqcap -(\textit{prim-EP } w\ v\ e))^\star$
using *6* **by** (*simp add*: *comp-associative*)
also have ... $= (\textit{prim-P } w\ v\ e)^T * (w \sqcap -(\textit{prim-EP } w\ v\ e))^+ * (w \sqcap -(\textit{prim-EP } w\ v\ e))^\star \sqcup (\textit{prim-P } w\ v\ e)^{T+} * (w \sqcap -(\textit{prim-EP } w\ v\ e))^\star$
by (*simp add*: *mult-left-dist-sup mult-right-dist-sup*)
also have ... $= (\textit{prim-P } w\ v\ e)^T * (w \sqcap -(\textit{prim-EP } w\ v\ e))^+ \sqcup (\textit{prim-P } w\ v\ e)^{T+} * (w \sqcap -(\textit{prim-EP } w\ v\ e))^\star$
by (*simp add*: *mult-assoc star.circ-transitive-equal*)
also have ... $= (\textit{prim-P } w\ v\ e)^T * (w \sqcap -(\textit{prim-EP } w\ v\ e))^+ \sqcup (\textit{prim-P } w\ v\ e)^{T+} * (1 \sqcup (w \sqcap -(\textit{prim-EP } w\ v\ e))^+)$
using *star-left-unfold-equal* **by** *simp*
also have ... $= (\textit{prim-P } w\ v\ e)^T * (w \sqcap -(\textit{prim-EP } w\ v\ e))^+ \sqcup (\textit{prim-P } w\ v\ e)^{T+} * (w \sqcap -(\textit{prim-EP } w\ v\ e))^+ \sqcup (\textit{prim-P } w\ v\ e)^{T+}$
by (*simp add*: *mult-left-dist-sup sup.left-commute sup-commute*)
also have ... $= ((\textit{prim-P } w\ v\ e)^T \sqcup (\textit{prim-P } w\ v\ e)^{T+}) * (w \sqcap -(\textit{prim-EP } w\ v\ e))^+ \sqcup (\textit{prim-P } w\ v\ e)^{T+}$
by (*simp add*: *mult-right-dist-sup*)
also have ... $= (\textit{prim-P } w\ v\ e)^{T+} * (w \sqcap -(\textit{prim-EP } w\ v\ e))^+ \sqcup (\textit{prim-P } w\ v\ e)^{T+}$
using *star.circ-mult-increasing* **by** (*simp add*: *le-iff-sup*)
also have ... ≤ -1
using *17 18* **by** *simp*
finally show *?thesis*
.
qed
have *19*: $e * (w \sqcap -(\textit{prim-EP } w\ v\ e))^\star * (\textit{prim-P } w\ v\ e)^{T\star} * (w \sqcap -(\textit{prim-EP } w\ v\ e))^\star \leq -1$
proof −
have $e * (w \sqcap -(\textit{prim-EP } w\ v\ e))^\star * (\textit{prim-P } w\ v\ e)^{T\star} * (w \sqcap -(\textit{prim-EP } w\ v\ e))^\star = e * ((w \sqcap -(\textit{prim-EP } w\ v\ e))^+ \sqcup (\textit{prim-P } w\ v\ e)^{T\star}) * (w \sqcap -(\textit{prim-EP}$

$w\ v\ e))^\star$
  using *6* **by** (*simp add*: *mult-assoc*)
 also have ... $= e * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^+ * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^\star \sqcup e * (prim\text{-}P\ w\ v\ e)^{T\star} * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^\star$
  by (*simp add*: *mult-left-dist-sup mult-right-dist-sup*)
 also have ... $= e * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^+ \sqcup e * (prim\text{-}P\ w\ v\ e)^{T\star} * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^\star$
  by (*simp add*: *mult-assoc star.circ-transitive-equal*)
 also have ... $\leq e * (prim\text{-}P\ w\ v\ e)^{T\star} * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^+ \sqcup e * (prim\text{-}P\ w\ v\ e)^{T\star} * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^\star$
  by (*metis mult-right-sub-dist-sup-right semiring.add-right-mono star.circ-back-loop-fixpoint*)
 also have ... $\leq e * (prim\text{-}P\ w\ v\ e)^{T\star} * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^\star$
  using *mult-right-isotone star.left-plus-below-circ* **by** *auto*
 also have ... $\leq v * -v^T$
  using *11* **by** *simp*
 also have ... ≤ -1
  by (*simp add*: *pp-increasing schroeder-3-p*)
 finally show *?thesis*
  .
qed
have *20*: $(prim\text{-}W\ w\ v\ e) * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^\star * (prim\text{-}P\ w\ v\ e)^{T\star} * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^\star \leq -1$
 using *15 16 19* **by** (*simp add*: *comp-right-dist-sup*)
have *21*: $(w \sqcap -(prim\text{-}EP\ w\ v\ e))^+ * e * (prim\text{-}P\ w\ v\ e)^{T\star} * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^\star \leq -1$
proof −
 have $(w \sqcap -(prim\text{-}EP\ w\ v\ e)) * v * -v^T \leq w * v * -v^T$
  by (*simp add*: *comp-isotone star-isotone*)
 also have ... $\leq v * -v^T$
  by (*simp add*: *assms*(*3*) *mult-left-isotone*)
 finally have *22*: $(w \sqcap -(prim\text{-}EP\ w\ v\ e)) * v * -v^T \leq v * -v^T$
  .
 have $(w \sqcap -(prim\text{-}EP\ w\ v\ e))^+ * e * (prim\text{-}P\ w\ v\ e)^{T\star} * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^\star \leq (w \sqcap -(prim\text{-}EP\ w\ v\ e))^+ * v * -v^T$
  using *11* **by** (*simp add*: *mult-right-isotone mult-assoc*)
 also have ... $\leq (w \sqcap -(prim\text{-}EP\ w\ v\ e))^\star * v * -v^T$
  using *mult-left-isotone star.left-plus-below-circ* **by** *blast*
 also have ... $\leq v * -v^T$
  using *22* **by** (*simp add*: *star-left-induct-mult mult-assoc*)
 also have ... ≤ -1
  by (*simp add*: *pp-increasing schroeder-3-p*)
 finally show *?thesis*
  .
qed
have *23*: $(prim\text{-}P\ w\ v\ e)^T * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^\star * e * (prim\text{-}P\ w\ v\ e)^{T\star} * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^\star \leq -1$
proof −
 have $(prim\text{-}P\ w\ v\ e)^T * (w \sqcap -(prim\text{-}EP\ w\ v\ e))^\star * e = (prim\text{-}P\ w\ v\ e)^T * e$

$\sqcup$ (*prim-P w v e*)^T $*$ ($w \sqcap -$(*prim-EP w v e*))$^+$ $*$ e
using *comp-left-dist-sup mult-assoc star.circ-loop-fixpoint sup-commute* **by** *auto*
also have ... $=$ *bot*
using *1 9* **by** *simp*
finally show *?thesis*
by *simp*
qed
have *24*: $e *$ ($w \sqcap -$(*prim-EP w v e*))* $* e *$ (*prim-P w v e*)^{T*} $*$ ($w \sqcap -$(*prim-EP w v e*))* ≤ -1
proof $-$
have $e *$ ($w \sqcap -$(*prim-EP w v e*))* $* e = e * e \sqcup e *$ ($w \sqcap -$(*prim-EP w v e*))$^+$ $* e$
using *comp-left-dist-sup mult-assoc star.circ-loop-fixpoint sup-commute* **by** *auto*
also have ... $=$ *bot*
using *2 10* **by** *simp*
finally show *?thesis*
by *simp*
qed
have *25*: (*prim-W w v e*) $*$ ($w \sqcap -$(*prim-EP w v e*))* $* e *$ (*prim-P w v e*)^{T*} $*$ ($w \sqcap -$(*prim-EP w v e*))* ≤ -1
using *21 23 24* **by** (*simp add*: *comp-right-dist-sup*)
have (*prim-W w v e*)* $=$ ((*prim-P w v e*)^T $\sqcup e$)* $*$ (($w \sqcap -$(*prim-EP w v e*)) $*$ ((*prim-P w v e*)^T $\sqcup e$)*)*
by (*metis star-sup-1 sup.left-commute sup-commute*)
also have ... $=$ ((*prim-P w v e*)^{T*} $\sqcup e *$ (*prim-P w v e*)^{T*}) $*$ (($w \sqcap -$(*prim-EP w v e*)) $*$ ((*prim-P w v e*)^{T*} $\sqcup e *$ (*prim-P w v e*)^{T*}))*
using *1 2 star-separate* **by** *auto*
also have ... $=$ ((*prim-P w v e*)^{T*} $\sqcup e *$ (*prim-P w v e*)^{T*}) $*$ (($w \sqcap -$(*prim-EP w v e*)) $*$ ($1 \sqcup e *$ (*prim-P w v e*)^{T*}))*
using *4 mult-left-dist-sup* **by** *auto*
also have ... $=$ ($w \sqcap -$(*prim-EP w v e*))* $*$ ((*prim-P w v e*)^{T*} $\sqcup e *$ (*prim-P w v e*)^{T*}) $*$ ($w \sqcap -$(*prim-EP w v e*))*
using *3 9 10 star-separate-2* **by** *blast*
also have ... $=$ ($w \sqcap -$(*prim-EP w v e*))* $*$ (*prim-P w v e*)^{T*} $*$ ($w \sqcap -$(*prim-EP w v e*))* $\sqcup$ ($w \sqcap -$(*prim-EP w v e*))* $* e *$ (*prim-P w v e*)^{T*} $*$ ($w \sqcap -$(*prim-EP w v e*))*
by (*simp add*: *semiring.distrib-left semiring.distrib-right mult-assoc*)
finally have (*prim-W w v e*)$^+$ $=$ (*prim-W w v e*) $*$ (($w \sqcap -$(*prim-EP w v e*))* $*$ (*prim-P w v e*)^{T*} $*$ ($w \sqcap -$(*prim-EP w v e*))* $\sqcup$ ($w \sqcap -$(*prim-EP w v e*))* $* e *$ (*prim-P w v e*)^{T*} $*$ ($w \sqcap -$(*prim-EP w v e*))*)
by *simp*
also have ... $=$ (*prim-W w v e*) $*$ ($w \sqcap -$(*prim-EP w v e*))* $*$ (*prim-P w v e*)^{T*} $*$ ($w \sqcap -$(*prim-EP w v e*))* $\sqcup$ (*prim-W w v e*) $*$ ($w \sqcap -$(*prim-EP w v e*))* $* e *$ (*prim-P w v e*)^{T*} $*$ ($w \sqcap -$(*prim-EP w v e*))*
by (*simp add*: *comp-left-dist-sup comp-associative*)
also have ... ≤ -1
using *20 25* **by** *simp*

finally show *?thesis*
.
qed

The following lemma shows that an edge across the cut between visited nodes and unvisited nodes does not leave the component of visited nodes.

lemma *mst-subgraph-inv*:
assumes $e \le v * -v^T \sqcap g$
and $t \le g$
and $v^T = r^T * t^\star$
shows $e \le (r^T * g^\star)^T * (r^T * g^\star) \sqcap g$
proof $-$
have $e \le v * -v^T \sqcap g$
by (*rule assms*(*1*))
also have ... $\le v * (-v^T \sqcap v^T * g) \sqcap g$
by (*simp add*: *dedekind-1*)
also have ... $\le v * v^T * g \sqcap g$
by (*simp add*: *comp-associative comp-right-isotone inf-commute le-infI2*)
also have ... $= v * (r^T * t^\star) * g \sqcap g$
by (*simp add*: *assms*(*3*))
also have ... $= (r^T * t^\star)^T * (r^T * t^\star) * g \sqcap g$
by (*metis assms*(*3*) *conv-involutive*)
also have ... $\le (r^T * t^\star)^T * (r^T * g^\star) * g \sqcap g$
using *assms*(*2*) *comp-inf.mult-left-isotone comp-isotone star-isotone* **by** *auto*
also have ... $\le (r^T * t^\star)^T * (r^T * g^\star) \sqcap g$
using *inf.sup-right-isotone inf-commute mult-assoc mult-right-isotone star.left-plus-below-circ star-plus* **by** *presburger*
also have ... $\le (r^T * g^\star)^T * (r^T * g^\star) \sqcap g$
using *assms*(*2*) *comp-inf.mult-left-isotone conv-dist-comp conv-isotone mult-left-isotone star-isotone* **by** *auto*
finally show *?thesis*
.
qed

The following lemmas show that the tree after exchanging contains the currently constructed and tree and its extension by the chosen edge.

lemma *mst-extends-old-tree*:
assumes $t \le w$
and $t \le v * v^T$
and *vector* v
shows $t \le$ *prim-W* w v e
proof $-$
have $t \sqcap$ *prim-EP* w v $e \le t \sqcap -v^T$
by (*simp add*: *inf.coboundedI2 inf.sup-monoid.add-assoc*)
also have ... $\le v * v^T \sqcap -v^T$
by (*simp add*: *assms*(*2*) *inf.coboundedI1*)
also have ... $\le$ *bot*
by (*simp add*: *assms*(*3*) *covector-vector-comp eq-refl schroeder-2*)
finally have $t \le -($*prim-EP* w v $e)$

using *le-bot pseudo-complement* **by** *blast*
hence $t \leq w \sqcap -(prim\text{-}EP\ w\ v\ e)$
using *assms*(*1*) **by** *simp*
thus *?thesis*
using *le-supI1* **by** *blast*
qed

lemma *mst-extends-new-tree*:
$t \leq w \Longrightarrow t \leq v * v^T \Longrightarrow vector\ v \Longrightarrow t \sqcup e \leq prim\text{-}W\ w\ v\ e$
using *mst-extends-old-tree* **by** *auto*

Lemmas *forests-bot-1*, *forests-bot-2*, *forests-bot-3* and *fc-comp-eq-fc* were contributed by Nicolas Robinson-O'Brien.

lemma *forests-bot-1*:
assumes *equivalence e*
and *forest f*
shows $(-e \sqcap f) * (e \sqcap f)^T = bot$
proof –
have $f * f^T \leq e$
using *assms dual-order.trans* **by** *blast*
hence $f * (e \sqcap f)^T \leq e$
by (*metis conv-dist-inf inf.boundedE inf.cobounded2 inf.orderE mult-right-isotone*)
hence $-e \sqcap f * (e \sqcap f)^T = bot$
by (*simp add*: *p-antitone pseudo-complement*)
thus *?thesis*
by (*metis assms*(*1*) *comp-isotone conv-dist-inf equivalence-comp-right-complement inf.boundedI inf.cobounded1 inf.cobounded2 le-bot*)
qed

lemma *forests-bot-2*:
assumes *equivalence e*
and *forest f*
shows $(-e \sqcap f^T) * x \sqcap (e \sqcap f^T) * y = bot$
proof –
have $(-e \sqcap f) * (e \sqcap f^T) = bot$
using *assms forests-bot-1 conv-dist-inf* **by** *simp*
thus *?thesis*
by (*smt assms*(*1*) *comp-associative comp-inf.semiring.mult-not-zero conv-complement conv-dist-comp conv-dist-inf conv-involutive dedekind-1 inf.cobounded2 inf.sup-monoid.add-commute le-bot mult-right-zero p-antitone-iff pseudo-complement semiring.mult-not-zero symmetric-top-closed top.extremum*)
qed

lemma *forests-bot-3*:
assumes *equivalence e*
and *forest f*
shows $x * (-e \sqcap f) \sqcap y * (e \sqcap f) = bot$

proof −
  have $(e \sqcap f) * (-e \sqcap f^T) = bot$
    using *assms forests-bot-1 conv-dist-inf conv-complement* **by** (*smt conv-dist-comp conv-involutive conv-order coreflexive-bot-closed coreflexive-symmetric*)
  hence $y * (e \sqcap f) * (-e \sqcap f^T) = bot$
    by (*simp add*: *comp-associative*)
  hence *1*: $x \sqcap y * (e \sqcap f) * (-e \sqcap f^T) = bot$
    using *comp-inf.semiring.mult-not-zero* **by** *blast*
  hence $(x \sqcap y * (e \sqcap f) * (-e \sqcap f^T)) * (-e \sqcap f) = bot$
    using *semiring.mult-not-zero* **by** *blast*
  hence $x * (-e \sqcap f^T)^T \sqcap y * (e \sqcap f) = bot$
    using *1 dedekind-2 inf-commute schroeder-2* **by** *auto*
  thus *?thesis*
    by (*simp add*: *assms*(*1*) *conv-complement conv-dist-inf*)
qed

lemma *acyclic-plus*:
  acyclic $x \Longrightarrow$ *acyclic* (x^+)
  by (*simp add*: *star.circ-transitive-equal star.left-plus-circ mult-assoc*)

end

We finally add the Kleene star to Stone relation algebras. Kleene star and the relational operations are reasonably independent. The only additional axiom we need in the generalisation to Stone-Kleene relation algebras is that star distributes over double complement.

class *stone-kleene-relation-algebra* = *stone-relation-algebra* + *pd-kleene-allegory* +
  assumes *pp-dist-star*: $--(x^\star) = (--x)^\star$
begin

lemma *reachable-without-loops*:
  $x^\star = (x \sqcap -1)^\star$
proof (*rule order.antisym*)
  have $x * (x \sqcap -1)^\star = (x \sqcap 1) * (x \sqcap -1)^\star \sqcup (x \sqcap -1) * (x \sqcap -1)^\star$
    by (*metis maddux-3-11-pp mult-right-dist-sup regular-one-closed*)
  also have ... $\leq (x \sqcap -1)^\star$
    by (*metis inf.cobounded2 le-supI mult-left-isotone star.circ-circ-mult star.left-plus-below-circ star-involutive star-one*)
  finally show $x^\star \leq (x \sqcap -1)^\star$
    by (*metis inf.cobounded2 maddux-3-11-pp regular-one-closed star.circ-circ-mult star.circ-sup-2 star-involutive star-sub-one*)
next
  show $(x \sqcap -1)^\star \leq x^\star$
    by (*simp add*: *star-isotone*)
qed

lemma *plus-reachable-without-loops*:
  $x^+ = (x \sqcap -1)^+ \sqcup (x \sqcap 1)$

by (*metis comp-associative maddux-3-11-pp regular-one-closed star.circ-back-loop-fixpoint star.circ-loop-fixpoint sup-assoc reachable-without-loops*)

lemma *star-plus-without-loops*:
$x^\star \sqcap -1 = x^+ \sqcap -1$
by (*metis maddux-3-13 star-left-unfold-equal*)

lemma *regular-closed-star*:
regular $x \Longrightarrow$ *regular* $(x^\star)$
by (*simp add*: *pp-dist-star*)

lemma *components-idempotent*:
components (*components* x) = *components* x
using *pp-dist-star star-involutive* **by** *auto*

lemma *fc-comp-eq-fc*:
$-$*forest-components* $(--f) = -$*forest-components* f
by (*metis conv-complement p-comp-pp p-pp-comp pp-dist-star*)

The following lemma shows that the nodes reachable in the tree after exchange contain the nodes reachable in the tree before exchange.

lemma *mst-reachable-inv*:
assumes *regular* (*prim-EP* w v e)
and *vector* r
and $e \leq v * -v^T$
and *vector* v
and $v^T = r^T * t^\star$
and $t \leq w$
and $t \leq v * v^T$
and $w * v \leq v$
shows $r^T * w^\star \leq r^T * (prim\text{-}W\ w\ v\ e)^\star$
proof $-$
have *1*: $r^T \leq r^T * (prim\text{-}W\ w\ v\ e)^\star$
using *sup.bounded-iff star.circ-back-loop-prefixpoint* **by** *blast*
have $top * e * (w^T \sqcap -v^T)^\star * w^T \sqcap -v^T = top * e * (w^T \sqcap -v^T)^\star * (w^T \sqcap -v^T)$
by (*simp add*: *assms*(*4*) *covector-comp-inf vector-conv-compl*)
also have ... $\leq top * e * (w^T \sqcap -v^T)^\star$
by (*simp add*: *comp-isotone mult-assoc star.circ-plus-same star.left-plus-below-circ*)
finally have *2*: $top * e * (w^T \sqcap -v^T)^\star * w^T \leq top * e * (w^T \sqcap -v^T)^\star \sqcup --v^T$
by (*simp add*: *shunting-var-p*)
have *3*: $--v^T * w^T \leq top * e * (w^T \sqcap -v^T)^\star \sqcup --v^T$
by (*metis assms*(*8*) *conv-dist-comp conv-order mult-assoc order.trans pp-comp-semi-commute pp-isotone sup.coboundedI1 sup-commute*)
have *4*: $top * e \leq top * e * (w^T \sqcap -v^T)^\star \sqcup --v^T$
using *sup-right-divisibility star.circ-back-loop-fixpoint le-supI1* **by** *blast*
have $(top * e * (w^T \sqcap -v^T)^\star \sqcup --v^T) * w^T = top * e * (w^T \sqcap -v^T)^\star * w^T \sqcup$

$--v^T * w^T$
  by ($simp\ add$: $comp\text{-}right\text{-}dist\text{-}sup$)
also have ... $\leq top * e * (w^T \sqcap -v^T)^\star \sqcup --v^T$
  using $2\ 3$ **by** $simp$
finally have $top * e \sqcup (top * e * (w^T \sqcap -v^T)^\star \sqcup --v^T) * w^T \leq top * e * (w^T \sqcap -v^T)^\star \sqcup --v^T$
  using 4 **by** $simp$
hence 5: $top * e * w^{T\star} \leq top * e * (w^T \sqcap -v^T)^\star \sqcup --v^T$
  by ($simp\ add$: $star\text{-}right\text{-}induct$)
have 6: $top * e \leq top * e * (w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star$
  using $sup\text{-}right\text{-}divisibility\ star.circ\text{-}back\text{-}loop\text{-}fixpoint$ **by** $blast$
have $(top * e * (w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star)^T \leq (top * e * w^{T\star})^T$
  by ($simp\ add$: $star\text{-}isotone\ mult\text{-}right\text{-}isotone\ conv\text{-}isotone\ inf\text{-}assoc$)
also have ... $= w^\star * e^T * top$
  by ($simp\ add$: $conv\text{-}dist\text{-}comp\ conv\text{-}star\text{-}commute\ mult\text{-}assoc$)
finally have 7: $(top * e * (w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star)^T \leq w^\star * e^T * top$
  .
have $(top * e * (w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star)^T \leq (top * e * (-v * -v^T)^\star)^T$
  by ($simp\ add$: $conv\text{-}isotone\ inf\text{-}commute\ mult\text{-}right\text{-}isotone\ star\text{-}isotone\ le\text{-}infI2$)
also have ... $\leq (top * v * -v^T * (-v * -v^T)^\star)^T$
  by ($metis\ assms(3)\ conv\text{-}isotone\ mult\text{-}left\text{-}isotone\ mult\text{-}right\text{-}isotone\ mult\text{-}assoc$)
also have ... $= (top * v * (-v^T * -v)^\star * -v^T)^T$
  by ($simp\ add$: $mult\text{-}assoc\ star\text{-}slide$)
also have ... $\leq (top * -v^T)^T$
  using $conv\text{-}order\ mult\text{-}left\text{-}isotone$ **by** $auto$
also have ... $= -v$
  by ($simp\ add$: $assms(4)\ conv\text{-}complement\ vector\text{-}conv\text{-}compl$)
finally have 8: $(top * e * (w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star)^T \leq w^\star * e^T * top \sqcap -v$
  using 7 **by** $simp$
have $covector\ (top * e * (w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star)$
  by ($simp\ add$: $covector\text{-}mult\text{-}closed$)
hence $top * e * (w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star * (w^T \sqcap -v^T) = top * e * (w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star * (w^T \sqcap -v^T \sqcap (top * e * (w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star)^T)$
  by ($metis\ comp\text{-}inf\text{-}vector\text{-}1\ inf.idem$)
also have ... $\leq top * e * (w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star * (w^T \sqcap -v^T \sqcap w^\star * e^T * top \sqcap -v)$
  using $8\ mult\text{-}right\text{-}isotone\ inf.sup\text{-}right\text{-}isotone\ inf\text{-}assoc$ **by** $simp$
also have ... $= top * e * (w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star * (w^T \sqcap (-v \sqcap -v^T) \sqcap w^\star * e^T * top)$
  using $inf\text{-}assoc\ inf\text{-}commute$ **by** ($simp\ add$: $inf\text{-}assoc$)
also have ... $= top * e * (w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star * (w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)$
  using $assms(4)\ conv\text{-}complement\ vector\text{-}complement\text{-}closed\ vector\text{-}covector$ **by**

fastforce
also have ... $\leq$ *top* $*$ *e* $*$ $(w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star$
by (*simp add*: *comp-associative comp-isotone star.circ-plus-same star.left-plus-below-circ*)
finally have *9*: *top* $*$ *e* $\sqcup$ *top* $*$ *e* $*$ $(w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star * (w^T \sqcap -v^T) \leq top * e * (w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star$
using *6* **by** *simp*
have *prim-EP w v e* $\leq -v^T \sqcap top * e * w^{T\star}$
using *inf.sup-left-isotone* **by** *auto*
also have ... $\leq$ *top* $*$ *e* $*$ $(w^T \sqcap -v^T)^\star$
using *5* **by** (*metis inf-commute shunting-var-p*)
also have ... $\leq$ *top* $*$ *e* $*$ $(w^T \sqcap -v * -v^T \sqcap w^\star * e^T * top)^\star$
using *9* **by** (*simp add*: *star-right-induct*)
finally have *10*: *prim-EP w v e* $\leq$ *top* $*$ *e* $*$ $(prim\text{-}P\ w\ v\ e)^{T\star}$
by (*simp add*: *conv-complement conv-dist-comp conv-dist-inf conv-star-commute mult-assoc*)
have *top* $*$ *e* $=$ *top* $* (v * -v^T \sqcap e)$
by (*simp add*: *assms*(*3*) *inf.absorb2*)
also have ... $\leq$ *top* $* (v * top \sqcap e)$
using *inf.sup-right-isotone inf-commute mult-right-isotone top-greatest* **by** *presburger*
also have ... $= (top \sqcap (v * top)^T) * e$
using *assms*(*4*) *covector-inf-comp-3* **by** *presburger*
also have ... $= top * v^T * e$
by (*simp add*: *conv-dist-comp*)
also have ... $= top * r^T * t^\star * e$
by (*simp add*: *assms*(*5*) *comp-associative*)
also have ... $\leq top * r^T * (prim\text{-}W\ w\ v\ e)^\star * e$
by (*metis assms*(*4*,*6*,*7*) *mst-extends-old-tree star-isotone mult-left-isotone mult-right-isotone*)
finally have *11*: $top * e \leq top * r^T * (prim\text{-}W\ w\ v\ e)^\star * e$
.
have $r^T * (prim\text{-}W\ w\ v\ e)^\star * (prim\text{-}EP\ w\ v\ e) \leq r^T * (prim\text{-}W\ w\ v\ e)^\star * (top * e * (prim\text{-}P\ w\ v\ e)^{T\star})$
using *10 mult-right-isotone* **by** *blast*
also have ... $= r^T * (prim\text{-}W\ w\ v\ e)^\star * top * e * (prim\text{-}P\ w\ v\ e)^{T\star}$
by (*simp add*: *mult-assoc*)
also have ... $\leq top * e * (prim\text{-}P\ w\ v\ e)^{T\star}$
by (*metis comp-associative comp-inf-covector inf.idem inf.sup-right-divisibility*)
also have ... $\leq top * r^T * (prim\text{-}W\ w\ v\ e)^\star * e * (prim\text{-}P\ w\ v\ e)^{T\star}$
using *11* **by** (*simp add*: *mult-left-isotone*)
also have ... $= r^T * (prim\text{-}W\ w\ v\ e)^\star * e * (prim\text{-}P\ w\ v\ e)^{T\star}$
using *assms*(*2*) *vector-conv-covector* **by** *auto*
also have ... $\leq r^T * (prim\text{-}W\ w\ v\ e)^\star * (prim\text{-}W\ w\ v\ e) * (prim\text{-}P\ w\ v\ e)^{T\star}$
by (*simp add*: *mult-left-isotone mult-right-isotone*)
also have ... $\leq r^T * (prim\text{-}W\ w\ v\ e)^\star * (prim\text{-}W\ w\ v\ e) * (prim\text{-}W\ w\ v\ e)^\star$
by (*meson dual-order.trans mult-right-isotone star-isotone sup-ge1 sup-ge2*)
also have ... $\leq r^T * (prim\text{-}W\ w\ v\ e)^\star$

by (*metis mult-assoc mult-right-isotone star.circ-transitive-equal star.left-plus-below-circ*)
finally have *12*: $r^T * (\textit{prim-W}\ w\ v\ e)^\star * (\textit{prim-EP}\ w\ v\ e) \leq r^T * (\textit{prim-W}\ w\ v\ e)^\star$
.
have $r^T * (\textit{prim-W}\ w\ v\ e)^\star * w \leq r^T * (\textit{prim-W}\ w\ v\ e)^\star * (w \sqcup \textit{prim-EP}\ w\ v\ e)$
by (*simp add*: *inf-assoc*)
also have ... $= r^T * (\textit{prim-W}\ w\ v\ e)^\star * ((w \sqcup \textit{prim-EP}\ w\ v\ e) \sqcap (-(\textit{prim-EP}\ w\ v\ e) \sqcup \textit{prim-EP}\ w\ v\ e))$
by (*metis assms*(*1*) *inf-top-right stone*)
also have ... $= r^T * (\textit{prim-W}\ w\ v\ e)^\star * ((w \sqcap -(\textit{prim-EP}\ w\ v\ e)) \sqcup \textit{prim-EP}\ w\ v\ e)$
by (*simp add*: *sup-inf-distrib2*)
also have ... $= r^T * (\textit{prim-W}\ w\ v\ e)^\star * (w \sqcap -(\textit{prim-EP}\ w\ v\ e)) \sqcup r^T * (\textit{prim-W}\ w\ v\ e)^\star * (\textit{prim-EP}\ w\ v\ e)$
by (*simp add*: *comp-left-dist-sup*)
also have ... $\leq r^T * (\textit{prim-W}\ w\ v\ e)^\star * (\textit{prim-W}\ w\ v\ e) \sqcup r^T * (\textit{prim-W}\ w\ v\ e)^\star * (\textit{prim-EP}\ w\ v\ e)$
using *mult-right-isotone sup-left-isotone* **by** *auto*
also have ... $\leq r^T * (\textit{prim-W}\ w\ v\ e)^\star \sqcup r^T * (\textit{prim-W}\ w\ v\ e)^\star * (\textit{prim-EP}\ w\ v\ e)$
using *mult-assoc mult-right-isotone star.circ-plus-same star.left-plus-below-circ sup-left-isotone* **by** *auto*
also have ... $= r^T * (\textit{prim-W}\ w\ v\ e)^\star$
using *12 sup.absorb1* **by** *blast*
finally have $r^T \sqcup r^T * (\textit{prim-W}\ w\ v\ e)^\star * w \leq r^T * (\textit{prim-W}\ w\ v\ e)^\star$
using *1* **by** *simp*
thus *?thesis*
by (*simp add*: *star-right-induct*)
qed

Some of the following lemmas already hold in pseudocomplemented distributive Kleene allegories.

4.1.3 Exchange gives Minimum Spanning Trees

The lemmas in this section are used to show that the after exchange we obtain a minimum spanning tree. The following lemmas show various interactions between the three constituents of the tree after exchange.

lemma *epm-1*:
vector $v \Longrightarrow \textit{prim-E}\ w\ v\ e \sqcup \textit{prim-P}\ w\ v\ e = \textit{prim-EP}\ w\ v\ e$
by (*metis inf-commute inf-sup-distrib1 mult-assoc mult-right-dist-sup regular-closed-p regular-complement-top vector-conv-compl*)

lemma *epm-2*:
assumes *regular* (*prim-EP* $w\ v\ e$)
and *vector* v
shows $(w \sqcap -(\textit{prim-EP}\ w\ v\ e)) \sqcup \textit{prim-P}\ w\ v\ e \sqcup \textit{prim-E}\ w\ v\ e = w$
proof –

have $(w \sqcap -(prim\text{-}EP\ w\ v\ e)) \sqcup prim\text{-}P\ w\ v\ e \sqcup prim\text{-}E\ w\ v\ e = (w \sqcap -(prim\text{-}EP\ w\ v\ e)) \sqcup prim\text{-}EP\ w\ v\ e$
using *epm-1 sup-assoc sup-commute assms*(*2*) **by** (*simp add*: *inf-sup-distrib1*)
also have ... $= w \sqcup prim\text{-}EP\ w\ v\ e$
by (*metis assms*(*1*) *inf-top.right-neutral regular-complement-top sup-inf-distrib2*)
also have ... $= w$
by (*simp add*: *sup-inf-distrib1*)
finally show *?thesis*
.
qed

lemma *epm-4*:
assumes $e \leq w$
and *injective* w
and $w * v \leq v$
and $e \leq v * -v^T$
shows $top * e * w^{T+} \leq top * v^T$
proof –
have $w^\star * v \leq v$
by (*simp add*: *assms*(*3*) *star-left-induct-mult*)
hence *1*: $v^T * w^{T\star} \leq v^T$
using *conv-star-commute conv-dist-comp conv-isotone* **by** *fastforce*
have $e * w^T \leq w * w^T \sqcap e * w^T$
by (*simp add*: *assms*(*1*) *mult-left-isotone*)
also have ... $\leq 1 \sqcap e * w^T$
using *assms*(*2*) *inf.sup-left-isotone* **by** *auto*
also have ... $= 1 \sqcap w * e^T$
using *calculation conv-dist-comp conv-involutive coreflexive-symmetric* **by** *fastforce*
also have ... $\leq w * e^T$
by *simp*
also have ... $\leq w * -v * v^T$
by (*metis assms*(*4*) *conv-complement conv-dist-comp conv-involutive conv-order mult-assoc mult-right-isotone*)
also have ... $\leq top * v^T$
by (*simp add*: *mult-left-isotone*)
finally have $top * e * w^{T+} \leq top * v^T * w^{T\star}$
by (*metis order.antisym comp-associative comp-isotone dense-top-closed mult-left-isotone transitive-top-closed*)
also have ... $\leq top * v^T$
using *1* **by** (*simp add*: *mult-assoc mult-right-isotone*)
finally show *?thesis*
.
qed

lemma *epm-5*:
assumes $e \leq w$
and *injective* w

and $w * v \le v$
and $e \le v * -v^T$
and *vector v*
shows *prim-P w v e* $= bot$
proof $-$
have *1*: $e = w \sqcap top * e$
by (*simp add*: *assms*(*1*,*2*) *epm-3*)
have *2*: $top * e * w^{T+} \le top * v^T$
by (*simp add*: *assms*(*1*−*4*) *epm-4*)
have *3*: $-v * -v^T \sqcap top * v^T = bot$
by (*simp add*: *assms*(*5*) *comp-associative covector-vector-comp inf.sup-monoid.add-commute schroeder-2*)
have *prim-P w v e* $= (w \sqcap -v * -v^T \sqcap top * e) \sqcup (w \sqcap -v * -v^T \sqcap top * e * w^{T+})$
by (*metis inf-sup-distrib1 mult-assoc star.circ-back-loop-fixpoint star-plus sup-commute*)
also have ... $\le (e \sqcap -v * -v^T) \sqcup (w \sqcap -v * -v^T \sqcap top * e * w^{T+})$
using *1* **by** (*metis comp-inf.mult-semi-associative inf.sup-monoid.add-commute semiring.add-right-mono*)
also have ... $\le (e \sqcap -v * -v^T) \sqcup (w \sqcap -v * -v^T \sqcap top * v^T)$
using *2* **by** (*metis sup-right-isotone inf.sup-right-isotone*)
also have ... $\le (e \sqcap -v * -v^T) \sqcup (-v * -v^T \sqcap top * v^T)$
using *inf.assoc le-infI2* **by** *auto*
also have ... $\le v * -v^T \sqcap -v * -v^T$
using *3 assms*(*4*) *inf.sup-left-isotone* **by** *auto*
also have ... $\le v * top \sqcap -v * top$
using *inf.sup-mono mult-right-isotone top-greatest* **by** *blast*
also have ... $= bot$
using *assms*(*5*) *inf-compl-bot vector-complement-closed* **by** *auto*
finally show *?thesis*
by (*simp add*: *le-iff-inf*)
qed

lemma *epm-6*:
assumes $e \le w$
and *injective w*
and $w * v \le v$
and $e \le v * -v^T$
and *vector v*
shows *prim-E w v e* $= e$
proof $-$
have *1*: $e \le --v * -v^T$
using *assms*(*4*) *mult-isotone order-lesseq-imp pp-increasing* **by** *blast*
have *2*: $top * e * w^{T+} \le top * v^T$
by (*simp add*: *assms*(*1*−*4*) *epm-4*)
have *3*: $e = w \sqcap top * e$
by (*simp add*: *assms*(*1*,*2*) *epm-3*)
hence $e \le top * e * w^{T\star}$
by (*metis le-infI2 star.circ-back-loop-fixpoint sup.commute sup-ge1*)

hence *4*: $e \le$ *prim-E w v e*
using *1* **by** (*simp add*: *assms*(*1*))
have *5*: $--v * -v^T \sqcap top * v^T = bot$
by (*simp add*: *assms*(*5*) *comp-associative covector-vector-comp inf.sup-monoid.add-commute schroeder-2*)
have *prim-E w v e* $= (w \sqcap --v * -v^T \sqcap top * e) \sqcup (w \sqcap --v * -v^T \sqcap top * e * w^{T+})$
by (*metis inf-sup-distrib1 mult-assoc star.circ-back-loop-fixpoint star-plus sup-commute*)
also have ... $\le (e \sqcap --v * -v^T) \sqcup (w \sqcap --v * -v^T \sqcap top * e * w^{T+})$
using *3* **by** (*metis comp-inf.mult-semi-associative inf.sup-monoid.add-commute semiring.add-right-mono*)
also have ... $\le (e \sqcap --v * -v^T) \sqcup (w \sqcap --v * -v^T \sqcap top * v^T)$
using *2* **by** (*metis sup-right-isotone inf.sup-right-isotone*)
also have ... $\le (e \sqcap --v * -v^T) \sqcup (--v * -v^T \sqcap top * v^T)$
using *inf.assoc le-infI2* **by** *auto*
also have ... $\le e$
by (*simp add*: *5*)
finally show *?thesis*
using *4* **by** (*simp add*: *order.antisym*)
qed

lemma *epm-7*:
regular (*prim-EP w v e*) $\Longrightarrow e \le w \Longrightarrow$ *injective* $w \Longrightarrow w * v \le v \Longrightarrow e \le v * -v^T \Longrightarrow$ *vector* $v \Longrightarrow$ *prim-W w v e* $= w$
by (*metis conv-bot epm-2 epm-5 epm-6*)

lemma *epm-8*:
assumes *acyclic w*
shows $(w \sqcap -($*prim-EP w v e*$)) \sqcap ($*prim-P w v e*$)^T = bot$
proof −
have $(w \sqcap -($*prim-EP w v e*$)) \sqcap ($*prim-P w v e*$)^T \le w \sqcap w^T$
by (*meson conv-isotone inf-le1 inf-mono order-trans*)
thus *?thesis*
by (*metis assms acyclic-asymmetric inf.commute le-bot*)
qed

lemma *epm-9*:
assumes $e \le v * -v^T$
and *vector v*
shows $(w \sqcap -($*prim-EP w v e*$)) \sqcap e = bot$
proof −
have *1*: $e \le -v^T$
by (*metis assms complement-conv-sub vector-conv-covector ev p-antitone-iff p-bot*)
have $(w \sqcap -($*prim-EP w v e*$)) \sqcap e = (w \sqcap --v^T \sqcap e) \sqcup (w \sqcap -(top * e * w^{T\star}) \sqcap e)$
by (*simp add*: *inf-commute inf-sup-distrib1*)
also have ... $\le (--v^T \sqcap e) \sqcup (-(top * e * w^{T\star}) \sqcap e)$

using *comp-inf.mult-left-isotone inf.cobounded2 semiring.add-mono* **by** *blast*
also have ... $= -(top * e * w^{T\star}) \sqcap e$
using *1* **by** (*metis inf.sup-relative-same-increasing inf-commute inf-sup-distrib1 maddux-3-13 regular-closed-p*)
also have ... $=$ *bot*
by (*metis inf.sup-relative-same-increasing inf-bot-right inf-commute inf-p mult-left-isotone star-outer-increasing top-greatest*)
finally show *?thesis*
by (*simp add*: *le-iff-inf*)
qed

lemma *epm-10*:
assumes $e \leq v * -v^T$
and *vector v*
shows $(prim\text{-}P\ w\ v\ e)^T \sqcap e = bot$
proof –
have $(prim\text{-}P\ w\ v\ e)^T \leq -v * -v^T$
by (*simp add*: *conv-complement conv-dist-comp conv-dist-inf inf.absorb-iff1 inf.left-commute inf-commute*)
hence $(prim\text{-}P\ w\ v\ e)^T \sqcap e \leq -v * -v^T \sqcap v * -v^T$
using *assms*(*1*) *inf-mono* **by** *blast*
also have ... $\leq -v * top \sqcap v * top$
using *inf.sup-mono mult-right-isotone top-greatest* **by** *blast*
also have ... $=$ *bot*
using *assms*(*2*) *inf-compl-bot vector-complement-closed* **by** *auto*
finally show *?thesis*
by (*simp add*: *le-iff-inf*)
qed

lemma *epm-11*:
assumes *vector v*
shows $(w \sqcap -(prim\text{-}EP\ w\ v\ e)) \sqcap prim\text{-}P\ w\ v\ e = bot$
proof –
have $prim\text{-}P\ w\ v\ e \leq prim\text{-}EP\ w\ v\ e$
by (*metis assms comp-isotone inf.sup-left-isotone inf.sup-right-isotone order.refl top-greatest vector-conv-compl*)
thus *?thesis*
using *inf-le2 order-trans p-antitone pseudo-complement* **by** *blast*
qed

lemma *epm-12*:
assumes *vector v*
shows $(w \sqcap -(prim\text{-}EP\ w\ v\ e)) \sqcap prim\text{-}E\ w\ v\ e = bot$
proof –
have $prim\text{-}E\ w\ v\ e \leq prim\text{-}EP\ w\ v\ e$
by (*metis assms comp-isotone inf.sup-left-isotone inf.sup-right-isotone order.refl top-greatest vector-conv-compl*)
thus *?thesis*
using *inf-le2 order-trans p-antitone pseudo-complement* **by** *blast*

qed

lemma *epm-13*:
assumes *vector v*
shows *prim-P w v e* $\sqcap$ *prim-E w v e* = *bot*
proof –
have *prim-P w v e* $\sqcap$ *prim-E w v e* $\leq -v * -v^T \sqcap --v * -v^T$
by (*meson dual-order.trans inf.coboundedl inf.sup-mono inf-le2*)
also have ... $\leq -v *$ *top* $\sqcap --v *$ *top*
using *inf.sup-mono mult-right-isotone top-greatest* **by** *blast*
also have ... = *bot*
using *assms inf-compl-bot vector-complement-closed* **by** *auto*
finally show *?thesis*
by (*simp add*: *le-iff-inf*)
qed

The following lemmas show that the relation characterising the edge across the cut is an arc.

lemma *arc-edge-1*:
assumes $e \leq v * -v^T \sqcap g$
and *vector v*
and $v^T = r^T * t^\star$
and $t \leq g$
and $r^T * g^\star \leq r^T * w^\star$
shows *top* $* e \leq v^T * w^\star$
proof –
have *top* $* e \leq$ *top* $* (v * -v^T \sqcap g)$
using *assms*(*1*) *mult-right-isotone* **by** *auto*
also have ... $\leq$ *top* $* (v *$ *top* $\sqcap g)$
using *inf.sup-right-isotone inf-commute mult-right-isotone top-greatest* **by** *presburger*
also have ... $= v^T * g$
by (*metis assms*(*2*) *covector-inf-comp-3 inf-top.left-neutral*)
also have ... $= r^T * t^\star * g$
by (*simp add*: *assms*(*3*))
also have ... $\leq r^T * g^\star * g$
by (*simp add*: *assms*(*4*) *mult-left-isotone mult-right-isotone star-isotone*)
also have ... $\leq r^T * g^\star$
by (*simp add*: *mult-assoc mult-right-isotone star.right-plus-below-circ*)
also have ... $\leq r^T * w^\star$
by (*simp add*: *assms*(*5*))
also have ... $\leq v^T * w^\star$
by (*metis assms*(*3*) *mult-left-isotone mult-right-isotone mult-1-right star.circ-reflexive*)
finally show *?thesis*
.
qed

lemma *arc-edge-2*:

assumes $e \leq v * -v^T \sqcap g$
  and *vector* v
  and $v^T = r^T * t^\star$
  and $t \leq g$
  and $r^T * g^\star \leq r^T * w^\star$
  and $w * v \leq v$
  and *injective* w
 shows $top * e * w^{T\star} \leq v^T * w^\star$
proof −
 have *1*: $top * e \leq v^T * w^\star$
  using *assms*(*1*−*5*) *arc-edge-1* **by** *blast*
 have $v^T * w^\star * w^T = v^T * w^T \sqcup v^T * w^+ * w^T$
  by (*metis mult-assoc mult-left-dist-sup star.circ-loop-fixpoint sup-commute*)
 also have ... $\leq v^T \sqcup v^T * w^+ * w^T$
  by (*metis assms*(*6*) *conv-dist-comp conv-isotone sup-left-isotone*)
 also have ... $= v^T \sqcup v^T * w^\star * (w * w^T)$
  by (*metis mult-assoc star-plus*)
 also have ... $\leq v^T \sqcup v^T * w^\star$
  by (*metis assms*(*7*) *mult-right-isotone mult-1-right sup-right-isotone*)
 also have ... $= v^T * w^\star$
  by (*metis star.circ-back-loop-fixpoint sup-absorb2 sup-ge2*)
 finally show *?thesis*
  using *1 star-right-induct* **by** *auto*
qed

lemma *arc-edge-3*:
 assumes $e \leq v * -v^T \sqcap g$
  and *vector* v
  and $v^T = r^T * t^\star$
  and $t \leq g$
  and $r^T * g^\star \leq r^T * w^\star$
  and $w * v \leq v$
  and *injective* w
  and *prim-E* $w\ v\ e = bot$
 shows $e = bot$
proof −
 have $bot =$ *prim-E* $w\ v\ e$
  by (*simp add*: *assms*(*8*))
 also have ... $= w \sqcap --v * top \sqcap top * -v^T \sqcap top * e * w^{T\star}$
  by (*metis assms*(*2*) *comp-inf-covector inf.assoc inf-top.left-neutral*
vector-conv-compl)
 also have ... $= w \sqcap top * e * w^{T\star} \sqcap -v^T \sqcap --v$
  using *assms*(*2*) *inf.assoc inf.commute vector-conv-compl*
vector-complement-closed **by** (*simp add*: *inf-assoc*)
 finally have *1*: $w \sqcap top * e * w^{T\star} \sqcap -v^T \leq -v$
  using *shunting-1-pp* **by** *force*
 have $w^\star * e^T * top = (top * e * w^{T\star})^T$
  by (*simp add*: *conv-star-commute comp-associative conv-dist-comp*)
 also have ... $\leq (v^T * w^\star)^T$

using *assms*(*1*−*7*) *arc-edge-2* **by** (*simp add*: *conv-isotone*)
also have ... $= w^{T\star} * v$
by (*simp add*: *conv-star-commute conv-dist-comp*)
finally have *2*: $w^\star * e^T * top \leq w^{T\star} * v$
.
have $(w^T \sqcap w^\star * e^T * top)^T * -v = (w \sqcap top * e * w^{T\star}) * -v$
by (*simp add*: *conv-dist-comp conv-dist-inf conv-star-commute mult-assoc*)
also have ... $= (w \sqcap top * e * w^{T\star} \sqcap -v^T) * top$
by (*metis assms*(*2*) *conv-complement covector-inf-comp-3 inf-top.right-neutral vector-complement-closed*)
also have ... $\leq -v * top$
using *1* **by** (*simp add*: *comp-isotone*)
also have ... $= -v$
using *assms*(*2*) *vector-complement-closed* **by** *auto*
finally have $(w^T \sqcap w^\star * e^T * top) * --v \leq --v$
using *p-antitone-iff schroeder-3-p* **by** *auto*
hence $w^\star * e^T * top \sqcap w^T * --v \leq --v$
by (*simp add*: *inf-vector-comp*)
hence *3*: $w^T * --v \leq --v \sqcup -(w^\star * e^T * top)$
by (*simp add*: *inf.commute shunting-p*)
have $w^T * -(w^\star * e^T * top) \leq -(w^\star * e^T * top)$
by (*metis mult-assoc p-antitone p-antitone-iff schroeder-3-p star.circ-loop-fixpoint sup-commute sup-right-divisibility*)
also have ... $\leq --v \sqcup -(w^\star * e^T * top)$
by *simp*
finally have $w^T * (--v \sqcup -(w^\star * e^T * top)) \leq --v \sqcup -(w^\star * e^T * top)$
using *3* **by** (*simp add*: *mult-left-dist-sup*)
hence $w^{T\star} * (--v \sqcup -(w^\star * e^T * top)) \leq --v \sqcup -(w^\star * e^T * top)$
using *star-left-induct-mult-iff* **by** *blast*
hence $w^{T\star} * --v \leq --v \sqcup -(w^\star * e^T * top)$
by (*simp add*: *semiring.distrib-left*)
hence $w^\star * e^T * top \sqcap w^{T\star} * --v \leq --v$
by (*simp add*: *inf-commute shunting-p*)
hence $w^\star * e^T * top \leq --v$
using *2* **by** (*metis inf.absorb1 p-antitone-iff p-comp-pp vector-export-comp*)
hence *4*: $e^T * top \leq --v$
by (*metis mult-assoc star.circ-loop-fixpoint sup.bounded-iff*)
have $e^T * top \leq (v * -v^T)^T * top$
using *assms*(*1*) *comp-isotone conv-isotone* **by** *auto*
also have ... $\leq -v * top$
by (*simp add*: *conv-complement conv-dist-comp mult-assoc mult-right-isotone*)
also have ... $= -v$
using *assms*(*2*) *vector-complement-closed* **by** *auto*
finally have $e^T * top \leq bot$
using *4* *shunting-1-pp* **by** *auto*
hence $e^T = bot$
using *order.antisym bot-least top-right-mult-increasing* **by** *blast*
thus *?thesis*
using *conv-bot* **by** *fastforce*

qed

lemma *arc-edge-4*:
assumes $e \leq v * -v^T \sqcap g$
and *vector v*
and $v^T = r^T * t^\star$
and $t \leq g$
and $r^T * g^\star \leq r^T * w^\star$
and *arc e*
shows $top * prim\text{-}E\ w\ v\ e * top = top$
proof $-$
have $--v^T * w = (--v^T * w \sqcap -v^T) \sqcup (--v^T * w \sqcap --v^T)$
by (*simp add*: *maddux-3-11-pp*)
also have ... $\leq (--v^T * w \sqcap -v^T) \sqcup --v^T$
using *sup-right-isotone* **by** *auto*
also have ... $= --v^T * (w \sqcap -v^T) \sqcup --v^T$
using *assms*(*2*) *covector-comp-inf covector-complement-closed vector-conv-covector* **by** *auto*
also have ... $\leq --v^T * (w \sqcap -v^T) * w^\star \sqcup --v^T$
by (*metis star.circ-back-loop-fixpoint sup.cobounded2 sup-left-isotone*)
finally have *1*: $--v^T * w \leq --v^T * (w \sqcap -v^T) * w^\star \sqcup --v^T$
.
have $--v^T * (w \sqcap -v^T) * w^\star * w \leq --v^T * (w \sqcap -v^T) * w^\star \sqcup --v^T$
by (*simp add*: *le-supI1 mult-assoc mult-right-isotone star.circ-plus-same star.left-plus-below-circ*)
hence *2*: $(--v^T * (w \sqcap -v^T) * w^\star \sqcup --v^T) * w \leq --v^T * (w \sqcap -v^T) * w^\star \sqcup --v^T$
using *1* **by** (*simp add*: *inf.orderE mult-right-dist-sup*)
have $v^T \leq --v^T * (w \sqcap -v^T) * w^\star \sqcup --v^T$
by (*simp add*: *pp-increasing sup.coboundedI2*)
hence $v^T * w^\star \leq --v^T * (w \sqcap -v^T) * w^\star \sqcup --v^T$
using *2* **by** (*simp add*: *star-right-induct*)
hence *3*: $-v^T \sqcap v^T * w^\star \leq --v^T * (w \sqcap -v^T) * w^\star$
by (*metis inf-commute shunting-var-p*)
have $top * e = top * e \sqcap v^T * w^\star$
by (*meson assms*(*1*$-$*5*) *arc-edge-1 inf.orderE*)
also have ... $\leq top * v * -v^T \sqcap v^T * w^\star$
using *assms*(*1*) *inf.sup-left-isotone mult-assoc mult-right-isotone* **by** *auto*
also have ... $\leq top * -v^T \sqcap v^T * w^\star$
using *inf.sup-left-isotone mult-left-isotone top-greatest* **by** *blast*
also have ... $= -v^T \sqcap v^T * w^\star$
by (*simp add*: *assms*(*2*) *vector-conv-compl*)
also have ... $\leq --v^T * (w \sqcap -v^T) * w^\star$
using *3* **by** *simp*
also have ... $= (top \sqcap (--v)^T) * (w \sqcap -v^T) * w^\star$
by (*simp add*: *conv-complement*)
also have ... $= top * (w \sqcap --v \sqcap -v^T) * w^\star$
using *assms*(*2*) *covector-inf-comp-3 inf-assoc inf-left-commute vector-complement-closed* **by** *presburger*

also have ... $= top * (w \sqcap --v * -v^T) * w^\star$
by (*metis assms*(*2*) *vector-complement-closed conv-complement inf-assoc vector-covector*)
finally have $top * (e^T * top)^T \leq top * (w \sqcap --v * -v^T) * w^\star$
by (*metis conv-dist-comp conv-involutive conv-top mult-assoc top-mult-top*)
hence $top \leq top * (w \sqcap --v * -v^T) * w^\star * (e^T * top)$
using *assms*(*6*) *shunt-bijective* **by** *blast*
also have ... $= top * (w \sqcap --v * -v^T) * (top * e * w^{\star T})^T$
by (*simp add*: *conv-dist-comp mult-assoc*)
also have ... $= top * (w \sqcap --v * -v^T \sqcap top * e * w^{\star T}) * top$
by (*simp add*: *comp-inf-vector-1 mult-assoc*)
finally show *?thesis*
by (*simp add*: *conv-star-commute top-le*)
qed

lemma *arc-edge-5*:
assumes *vector v*
and $w * v \leq v$
and *injective w*
and *arc e*
shows $(prim\text{-}E\ w\ v\ e)^T * top * prim\text{-}E\ w\ v\ e \leq 1$
proof −
have *1*: $e^T * top * e \leq 1$
by (*simp add*: *assms*(*4*) *point-injective*)
have $prim\text{-}E\ w\ v\ e \leq --v * top$
by (*simp add*: *inf-commute le-infI2 mult-right-isotone*)
hence *2*: $prim\text{-}E\ w\ v\ e \leq --v$
by (*simp add*: *assms*(*1*) *vector-complement-closed*)
have *3*: $w * --v \leq --v$
by (*simp add*: *assms*(*2*) *p-antitone p-antitone-iff*)
have $w \sqcap top * prim\text{-}E\ w\ v\ e \leq w * (prim\text{-}E\ w\ v\ e)^T * prim\text{-}E\ w\ v\ e$
by (*metis dedekind-2 inf.commute inf-top.left-neutral*)
also have ... $\leq w * w^T * prim\text{-}E\ w\ v\ e$
by (*simp add*: *conv-isotone le-infI1 mult-left-isotone mult-right-isotone*)
also have ... $\leq prim\text{-}E\ w\ v\ e$
by (*metis assms*(*3*) *mult-left-isotone mult-left-one*)
finally have *4*: $w \sqcap top * prim\text{-}E\ w\ v\ e \leq prim\text{-}E\ w\ v\ e$
.
have $w^+ \sqcap top * prim\text{-}E\ w\ v\ e = w^\star * (w \sqcap top * prim\text{-}E\ w\ v\ e)$
by (*simp add*: *comp-inf-covector star-plus*)
also have ... $\leq w^\star * prim\text{-}E\ w\ v\ e$
using *4* **by** (*simp add*: *mult-right-isotone*)
also have ... $\leq --v$
using *2 3 star-left-induct sup.bounded-iff* **by** *blast*
finally have *5*: $w^+ \sqcap top * prim\text{-}E\ w\ v\ e \sqcap -v = bot$
using *shunting-1-pp* **by** *blast*
hence *6*: $w^{+T} \sqcap (prim\text{-}E\ w\ v\ e)^T * top \sqcap -v^T = bot$
using *conv-complement conv-dist-comp conv-dist-inf conv-top conv-bot* **by** *force*

have $(prim\text{-}E\ w\ v\ e)^T * top * prim\text{-}E\ w\ v\ e \leq (top * e * w^{T\star})^T * top * (top * e * w^{T\star})$
by (*simp add*: *conv-isotone mult-isotone*)
also have ... $= w^\star * e^T * top * e * w^{T\star}$
by (*metis conv-star-commute conv-dist-comp conv-involutive conv-top mult-assoc top-mult-top*)
also have ... $\leq w^\star * w^{T\star}$
using *1* **by** (*metis mult-assoc mult-1-right mult-right-isotone mult-left-isotone*)
also have ... $= w^\star \sqcup w^{T\star}$
by (*metis assms*(*3*) *cancel-separate order.eq-iff star.circ-sup-sub-sup-one-1 star.circ-plus-one star-involutive*)
also have ... $= w^+ \sqcup w^{T+} \sqcup 1$
by (*metis star.circ-plus-one star-left-unfold-equal sup.assoc sup.commute*)
finally have *7*: $(prim\text{-}E\ w\ v\ e)^T * top * prim\text{-}E\ w\ v\ e \leq w^+ \sqcup w^{T+} \sqcup 1$
.
have $prim\text{-}E\ w\ v\ e \leq --v * -v^T$
by (*simp add*: *le-infI1*)
also have ... $\leq top * -v^T$
by (*simp add*: *mult-left-isotone*)
also have ... $= -v^T$
by (*simp add*: *assms*(*1*) *vector-conv-compl*)
finally have *8*: $prim\text{-}E\ w\ v\ e \leq -v^T$
.
hence *9*: $(prim\text{-}E\ w\ v\ e)^T \leq -v$
by (*metis conv-complement conv-involutive conv-isotone*)
have $(prim\text{-}E\ w\ v\ e)^T * top * prim\text{-}E\ w\ v\ e = (w^+ \sqcup w^{T+} \sqcup 1) \sqcap (prim\text{-}E\ w\ v\ e)^T * top * prim\text{-}E\ w\ v\ e$
using *7* **by** (*simp add*: *inf.absorb-iff2*)
also have ... $= (1 \sqcap (prim\text{-}E\ w\ v\ e)^T * top * prim\text{-}E\ w\ v\ e) \sqcup (w^+ \sqcap (prim\text{-}E\ w\ v\ e)^T * top * prim\text{-}E\ w\ v\ e) \sqcup (w^{T+} \sqcap (prim\text{-}E\ w\ v\ e)^T * top * prim\text{-}E\ w\ v\ e)$
using *comp-inf.mult-right-dist-sup sup-assoc sup-commute* **by** *auto*
also have ... $\leq 1 \sqcup (w^+ \sqcap (prim\text{-}E\ w\ v\ e)^T * top * prim\text{-}E\ w\ v\ e) \sqcup (w^{T+} \sqcap (prim\text{-}E\ w\ v\ e)^T * top * prim\text{-}E\ w\ v\ e)$
using *inf-le1 sup-left-isotone* **by** *blast*
also have ... $\leq 1 \sqcup (w^+ \sqcap (prim\text{-}E\ w\ v\ e)^T * top * prim\text{-}E\ w\ v\ e) \sqcup (w^{T+} \sqcap (prim\text{-}E\ w\ v\ e)^T * top * -v^T)$
using *8 inf.sup-right-isotone mult-right-isotone sup-right-isotone* **by** *blast*
also have ... $\leq 1 \sqcup (w^+ \sqcap -v * top * prim\text{-}E\ w\ v\ e) \sqcup (w^{T+} \sqcap (prim\text{-}E\ w\ v\ e)^T * top * -v^T)$
using *9* **by** (*metis inf.sup-right-isotone mult-left-isotone sup.commute sup-right-isotone*)
also have ... $= 1 \sqcup (w^+ \sqcap -v * top \sqcap top * prim\text{-}E\ w\ v\ e) \sqcup (w^{T+} \sqcap (prim\text{-}E\ w\ v\ e)^T * top \sqcap top * -v^T)$
by (*metis* (*no-types*) *vector-export-comp inf-top-right inf-assoc*)
also have ... $= 1 \sqcup (w^+ \sqcap -v \sqcap top * prim\text{-}E\ w\ v\ e) \sqcup (w^{T+} \sqcap (prim\text{-}E\ w\ v\ e)^T * top \sqcap -v^T)$
using *assms*(*1*) *vector-complement-closed vector-conv-compl* **by** *auto*
also have ... $= 1$
using *5 6* **by** (*simp add*: *conv-star-commute conv-dist-comp inf.commute*

inf-assoc star.circ-plus-same)
  finally show *?thesis*
    .
qed

lemma *arc-edge-6*:
  assumes *vector v*
      and $w * v \leq v$
      and *injective w*
      and *arc e*
    shows *prim-E w v e* $*$ *top* $*$ (*prim-E w v e*)$^T \leq$ *1*
proof −
  have *prim-E w v e* $*$ *1* $*$ (*prim-E w v e*)$^T \leq w * w^T$
    using *comp-isotone conv-order inf.coboundedI1 mult-one-associative* **by** *auto*
  also have ... $\leq$ *1*
    by (*simp add*: *assms*(*3*))
  finally have *1*: *prim-E w v e* $*$ *1* $*$ (*prim-E w v e*)$^T \leq$ *1*
    .
  have (*prim-E w v e*)^T $*$ *top* $*$ *prim-E w v e* $\leq$ *1*
    by (*simp add*: *assms arc-edge-5*)
  also have ... $\leq --$*1*
    by (*simp add*: *pp-increasing*)
  finally have *2*: *prim-E w v e* $* -$*1* $*$ (*prim-E w v e*)$^T \leq$ *bot*
    by (*metis conv-involutive regular-closed-bot regular-dense-top triple-schroeder-p*)
  have *prim-E w v e* $*$ *top* $*$ (*prim-E w v e*)^T = *prim-E w v e* $*$ *1* $*$ (*prim-E w v e*)$^T \sqcup$ *prim-E w v e* $* -$*1* $*$ (*prim-E w v e*)^T
    by (*metis mult-left-dist-sup mult-right-dist-sup regular-complement-top regular-one-closed*)
  also have ... $\leq$ *1*
    using *1 2* **by** (*simp add*: *bot-unique*)
  finally show *?thesis*
    .
qed

lemma *arc-edge*:
  assumes $e \leq v * -v^T \sqcap g$
      and *vector v*
      and $v^T = r^T * t^\star$
      and $t \leq g$
      and $r^T * g^\star \leq r^T * w^\star$
      and $w * v \leq v$
      and *injective w*
      and *arc e*
    shows *arc* (*prim-E w v e*)
proof (*intro conjI*)
  have *prim-E w v e* $*$ *top* $*$ (*prim-E w v e*)$^T \leq$ *1*
    using *assms*(*2*,*6*−*8*) *arc-edge-6* **by** *simp*
  thus *injective* (*prim-E w v e* $*$ *top*)

by (*metis conv-dist-comp conv-top mult-assoc top-mult-top*)
next
show *surjective* (*prim-E w v e* $*$ *top*)
using *assms*(1−5,8) *arc-edge-4 mult-assoc* **by** *simp*
next
have $(\mathit{prim\text{-}E}\ w\ v\ e)^T * \mathit{top} * \mathit{prim\text{-}E}\ w\ v\ e \leq 1$
using *assms*(2,6−8) *arc-edge-5* **by** *simp*
thus *injective* $((\mathit{prim\text{-}E}\ w\ v\ e)^T * \mathit{top})$
by (*metis conv-dist-comp conv-involutive conv-top mult-assoc top-mult-top*)
next
have $\mathit{top} * \mathit{prim\text{-}E}\ w\ v\ e * \mathit{top} = \mathit{top}$
using *assms*(1−5,8) *arc-edge-4* **by** *simp*
thus *surjective* $((\mathit{prim\text{-}E}\ w\ v\ e)^T * \mathit{top})$
by (*metis mult-assoc conv-dist-comp conv-top*)
qed

4.1.4 Invariant implies Postcondition

The lemmas in this section are used to show that the invariant implies the postcondition at the end of the algorithm. The following lemma shows that the nodes reachable in the graph are the same as those reachable in the constructed tree.

lemma *span-post*:
assumes *regular v*
and *vector v*
and $v^T = r^T * t^\star$
and $v * -v^T \sqcap g = \mathit{bot}$
and $t \leq v * v^T \sqcap g$
and $r^T * (v * v^T \sqcap g)^\star \leq r^T * t^\star$
shows $v^T = r^T * g^\star$
proof −
let $?vv = v * v^T \sqcap g$
have 1: $r^T \leq v^T$
using *assms*(3) *mult-right-isotone mult-1-right star.circ-reflexive* **by** *fastforce*
have $v * \mathit{top} \sqcap g = (v * v^T \sqcup v * -v^T) \sqcap g$
by (*metis assms*(1) *conv-complement mult-left-dist-sup regular-complement-top*)
also have ... $= ?vv \sqcup (v * -v^T \sqcap g)$
by (*simp add*: *inf-sup-distrib2*)
also have ... $= ?vv$
by (*simp add*: *assms*(4))
finally have 2: $v * \mathit{top} \sqcap g = ?vv$
by *simp*
have $r^T * ?vv^\star \leq v^T * ?vv^\star$
using *1* **by** (*simp add*: *comp-left-isotone*)
also have ... $\leq v^T * (v * v^T)^\star$
by (*simp add*: *comp-right-isotone star.circ-isotone*)
also have ... $\leq v^T$
by (*simp add*: *assms*(2) *vector-star-1*)

finally have $r^T * ?vv^\star \leq v^T$
  by *simp*
hence $r^T * ?vv^\star * g = (r^T * ?vv^\star \sqcap v^T) * g$
  by (*simp add*: *inf.absorb1*)
also have ... $= r^T * ?vv^\star * (v * top \sqcap g)$
  by (*simp add*: *assms*(*2*) *covector-inf-comp-3*)
also have ... $= r^T * ?vv^\star * ?vv$
  using *2* **by** *simp*
also have ... $\leq r^T * ?vv^\star$
  by (*simp add*: *comp-associative comp-right-isotone star.left-plus-below-circ star-plus*)
finally have $r^T \sqcup r^T * ?vv^\star * g \leq r^T * ?vv^\star$
  using *star.circ-back-loop-prefixpoint* **by** *auto*
hence $r^T * g^\star \leq r^T * ?vv^\star$
  using *star-right-induct* **by** *blast*
hence $r^T * g^\star = r^T * ?vv^\star$
  by (*simp add*: *order.antisym mult-right-isotone star-isotone*)
also have ... $= r^T * t^\star$
  using *assms*(*5*,*6*) *order.antisym mult-right-isotone star-isotone* **by** *auto*
also have ... $= v^T$
  by (*simp add*: *assms*(*3*))
finally show *?thesis*
  by *simp*
qed

The following lemma shows that the minimum spanning tree extending a tree is the same as the tree at the end of the algorithm.

lemma *mst-post*:
  assumes *vector* r
    and *injective* r
    and $v^T = r^T * t^\star$
    and *forest* w
    and $t \leq w$
    and $w \leq v * v^T$
  shows $w = t$
proof –
have *1*: *vector* v
  using *assms*(*1*,*3*) *covector-mult-closed vector-conv-covector* **by** *auto*
have $w * v \leq v * v^T * v$
  by (*simp add*: *assms*(*6*) *mult-left-isotone*)
also have ... $\leq v$
  using *1* **by** (*metis mult-assoc mult-right-isotone top-greatest*)
finally have *2*: $w * v \leq v$
  .
have *3*: $r \leq v$
  by (*metis assms*(*3*) *conv-order mult-right-isotone mult-1-right star.circ-reflexive*)
have *4*: $v \sqcap -r = t^{T\star} * r \sqcap -r$
  by (*metis assms*(*3*) *conv-dist-comp conv-involutive conv-star-commute*)

also have ... $= (r \sqcup t^{T+} * r) \sqcap -r$
using *mult-assoc star.circ-loop-fixpoint sup-commute* **by** *auto*
also have ... $\leq t^{T+} * r$
by (*simp add*: *shunting*)
also have ... $\leq t^{T} * top$
by (*simp add*: *comp-isotone mult-assoc*)
finally have $1 \sqcap (v \sqcap -r) * (v \sqcap -r)^{T} \leq 1 \sqcap t^{T} * top * (t^{T} * top)^{T}$
using *conv-order inf.sup-right-isotone mult-isotone* **by** *auto*
also have ... $= 1 \sqcap t^{T} * top * t$
by (*metis conv-dist-comp conv-involutive conv-top mult-assoc top-mult-top*)
also have ... $\leq t^{T} * (top * t \sqcap t * 1)$
by (*metis conv-involutive dedekind-1 inf.commute mult-assoc*)
also have ... $\leq t^{T} * t$
by (*simp add*: *mult-right-isotone*)
finally have *5*: $1 \sqcap (v \sqcap -r) * (v \sqcap -r)^{T} \leq t^{T} * t$
.
have $w * w^{+} \leq -1$
by (*metis assms*(*4*) *mult-right-isotone order-trans star.circ-increasing star.left-plus-circ*)
hence *6*: $w^{T+} \leq -w$
by (*metis conv-star-commute mult-assoc mult-1-left triple-schroeder-p*)
have $w * r \sqcap w^{T+} * r = (w \sqcap w^{T+}) * r$
using *assms*(*2*) **by** (*simp add*: *injective-comp-right-dist-inf*)
also have ... $= bot$
using *6 p-antitone pseudo-complement-pp semiring.mult-not-zero* **by** *blast*
finally have *7*: $w * r \sqcap w^{T+} * r = bot$
.
have $-1 * r \leq -r$
using *assms*(*2*) *dual-order.trans pp-increasing schroeder-4-p* **by** *blast*
hence $-1 * r * top \leq -r$
by (*simp add*: *assms*(*1*) *comp-associative*)
hence *8*: $r^{T} * -1 * r \leq bot$
by (*simp add*: *mult-assoc schroeder-6-p*)
have $r^{T} * w * r \leq r^{T} * w^{+} * r$
by (*simp add*: *mult-left-isotone mult-right-isotone star.circ-mult-increasing*)
also have ... $\leq r^{T} * -1 * r$
by (*simp add*: *assms*(*4*) *comp-isotone*)
finally have $r^{T} * w * r \leq bot$
using *8* **by** *simp*
hence $w * r * top \leq -r$
by (*simp add*: *mult-assoc schroeder-6-p*)
hence $w * r \leq -r$
by (*simp add*: *assms*(*1*) *comp-associative*)
hence $w * r \leq -r \sqcap w * v$
using *3* **by** (*simp add*: *mult-right-isotone*)
also have ... $\leq -r \sqcap v$
using *2* **by** (*simp add*: *le-infI2*)
also have ... $= -r \sqcap t^{T\star} * r$
using *4* **by** (*simp add*: *inf-commute*)

also have ... $\leq -r \sqcap w^{T\star} * r$
using *assms*(*5*) *comp-inf.mult-right-isotone conv-isotone mult-left-isotone star-isotone* **by** *auto*
also have ... $= -r \sqcap (r \sqcup w^{T+} * r)$
using *mult-assoc star.circ-loop-fixpoint sup-commute* **by** *auto*
also have ... $\leq w^{T+} * r$
using *inf.commute maddux-3-13* **by** *auto*
finally have $w * r = bot$
using *7* **by** (*simp add*: *le-iff-inf*)
hence $w = w \sqcap top * -r^T$
by (*metis complement-conv-sub conv-dist-comp conv-involutive conv-bot inf.assoc inf.orderE regular-closed-bot regular-dense-top top-left-mult-increasing*)
also have ... $= w \sqcap v * v^T \sqcap top * -r^T$
by (*simp add*: *assms*(*6*) *inf-absorb1*)
also have ... $\leq w \sqcap top * v^T \sqcap top * -r^T$
using *comp-inf.mult-left-isotone comp-inf.mult-right-isotone mult-left-isotone* **by** *auto*
also have ... $= w \sqcap top * (v^T \sqcap -r^T)$
using *1 assms*(*1*) *covector-inf-closed inf-assoc vector-conv-compl vector-conv-covector* **by** *auto*
also have ... $= w * (1 \sqcap (v \sqcap -r) * top)$
by (*simp add*: *comp-inf-vector conv-complement conv-dist-inf*)
also have ... $= w * (1 \sqcap (v \sqcap -r) * (v \sqcap -r)^T)$
by (*metis conv-top dedekind-eq inf-commute inf-top-left mult-1-left mult-1-right*)
also have ... $\leq w * t^T * t$
using *5* **by** (*simp add*: *comp-isotone mult-assoc*)
also have ... $\leq w * w^T * t$
by (*simp add*: *assms*(*5*) *comp-isotone conv-isotone*)
also have ... $\leq t$
using *assms*(*4*) *mult-left-isotone mult-1-left* **by** *fastforce*
finally show *?thesis*
by (*simp add*: *assms*(*5*) *order.antisym*)
qed

4.2 Kruskal's Algorithm

The following results are used for proving the correctness of Kruskal's minimum spanning tree algorithm.

4.2.1 Preservation of Invariant

We first treat the preservation of the invariant. The following lemmas show conditions necessary for preserving that f is a forest.

lemma *kruskal-injective-inv-2*:
assumes *arc e*
and *acyclic f*
shows $top * e * f^{T\star} * f^T \leq -e$

proof −
  have $f \le -f^{T\star}$
    using *assms*(*2*) *acyclic-star-below-complement p-antitone-iff* **by** *simp*
  hence $e * f \le top * e * -f^{T\star}$
    by (*simp add*: *comp-isotone top-left-mult-increasing*)
  also have ... $= -(top * e * f^{T\star})$
    by (*metis assms*(*1*) *comp-mapping-complement conv-dist-comp conv-involutive conv-top*)
  finally show *?thesis*
    using *schroeder-4-p* **by** *simp*
qed

lemma *kruskal-injective-inv-3*:
  assumes *arc e*
      and *forest f*
    shows $(top * e * f^{T\star})^T * (top * e * f^{T\star}) \sqcap f^T * f \le 1$
proof −
  have $(top * e * f^{T\star})^T * (top * e * f^{T\star}) = f^\star * e^T * top * e * f^{T\star}$
    by (*metis conv-dist-comp conv-involutive conv-star-commute conv-top vector-top-closed mult-assoc*)
  also have ... $\le f^\star * f^{T\star}$
    by (*metis assms*(*1*) *arc-expanded mult-left-isotone mult-right-isotone mult-1-left mult-assoc*)
  finally have $(top * e * f^{T\star})^T * (top * e * f^{T\star}) \sqcap f^T * f \le f^\star * f^{T\star} \sqcap f^T * f$
    using *inf.sup-left-isotone* **by** *simp*
  also have ... ≤ 1
    using *assms*(*2*) *forest-separate* **by** *simp*
  finally show *?thesis*
    by *simp*
qed

lemma *kruskal-acyclic-inv*:
  assumes *acyclic f*
      and *covector q*
      and $(f \sqcap q)^T * f^\star * e = bot$
      and $e * f^\star * e = bot$
      and $f^{T\star} * f^\star \le -e$
    shows *acyclic* $((f \sqcap -q) \sqcup (f \sqcap q)^T \sqcup e)$
proof −
  have $(f \sqcap -q) * (f \sqcap q)^T = (f \sqcap -q) * (f^T \sqcap q^T)$
    by (*simp add*: *conv-dist-inf*)
  hence *1*: $(f \sqcap -q) * (f \sqcap q)^T = bot$
    by (*metis assms*(*2*) *comp-inf.semiring.mult-zero-right comp-inf-vector-1 conv-bot covector-bot-closed inf.sup-monoid.add-assoc p-inf*)
  hence *2*: $(f \sqcap -q)^\star * (f \sqcap q)^T = (f \sqcap q)^T$
    using *mult-right-zero star-absorb star-simulation-right-equal* **by** *fastforce*
  hence *3*: $((f \sqcap -q) \sqcup (f \sqcap q)^T)^+ = (f \sqcap q)^{T\star} * (f \sqcap -q)^+ \sqcup (f \sqcap q)^{T+}$
    by (*simp add*: *plus-sup*)
  have *4*: $((f \sqcap -q) \sqcup (f \sqcap q)^T)^\star = (f \sqcap q)^{T\star} * (f \sqcap -q)^\star$

using *2* **by** (*simp add*: *star.circ-sup-9*)
have $(f \sqcap q)^T * (f \sqcap -q)^\star * e \le (f \sqcap q)^T * f^\star * e$
by (*simp add*: *mult-left-isotone mult-right-isotone star-isotone*)
hence $(f \sqcap q)^T * (f \sqcap -q)^\star * e = bot$
using *assms*(*3*) *le-bot* **by** *simp*
hence *5*: $(f \sqcap q)^{T\star} * (f \sqcap -q)^\star * e = (f \sqcap -q)^\star * e$
by (*metis comp-associative conv-bot conv-dist-comp conv-involutive conv-star-commute star-absorb*)
have $e * (f \sqcap -q)^\star * e \le e * f^\star * e$
by (*simp add*: *mult-left-isotone mult-right-isotone star-isotone*)
hence $e * (f \sqcap -q)^\star * e = bot$
using *assms*(*4*) *le-bot* **by** *simp*
hence *6*: $((f \sqcap -q)^\star * e)^+ = (f \sqcap -q)^\star * e$
by (*simp add*: *comp-associative star-absorb*)
have $f^{T\star} * 1 * f^{T\star} * f^\star \le -e$
by (*simp add*: *assms*(*5*) *star.circ-transitive-equal*)
hence *7*: $f^\star * e * f^{T\star} * f^\star \le -1$
by (*metis comp-right-one conv-involutive conv-one conv-star-commute triple-schroeder-p*)
have $(f \sqcap -q)^+ * (f \sqcap q)^{T+} \le -1$
using *1 2* **by** (*metis forest-bot mult-left-zero mult-assoc*)
hence *8*: $(f \sqcap q)^{T+} * (f \sqcap -q)^+ \le -1$
using *comp-commute-below-diversity* **by** *simp*
have *9*: $f^{T+} \le -1$
using *assms*(*1*) *acyclic-star-below-complement schroeder-5-p* **by** *force*
have $((f \sqcap -q) \sqcup (f \sqcap q)^T \sqcup e)^+ = (((f \sqcap -q) \sqcup (f \sqcap q)^T)^\star * e)^\star * ((f \sqcap -q) \sqcup (f \sqcap q)^T)^+ \sqcup (((f \sqcap -q) \sqcup (f \sqcap q)^T)^\star * e)^+$
by (*simp add*: *plus-sup*)
also have ... $= ((f \sqcap q)^{T\star} * (f \sqcap -q)^\star * e)^\star * ((f \sqcap q)^{T\star} * (f \sqcap -q)^+ \sqcup (f \sqcap q)^{T+}) \sqcup ((f \sqcap q)^{T\star} * (f \sqcap -q)^\star * e)^+$
using *3 4* **by** *simp*
also have ... $= ((f \sqcap -q)^\star * e)^\star * ((f \sqcap q)^{T\star} * (f \sqcap -q)^+ \sqcup (f \sqcap q)^{T+}) \sqcup ((f \sqcap -q)^\star * e)^+$
using *5* **by** *simp*
also have ... $= ((f \sqcap -q)^\star * e \sqcup 1) * ((f \sqcap q)^{T\star} * (f \sqcap -q)^+ \sqcup (f \sqcap q)^{T+}) \sqcup (f \sqcap -q)^\star * e$
using *6* **by** (*metis star-left-unfold-equal sup-monoid.add-commute*)
also have ... $= (f \sqcap -q)^\star * e \sqcup (f \sqcap -q)^\star * e * (f \sqcap q)^{T+} \sqcup (f \sqcap -q)^\star * e * (f \sqcap q)^{T\star} * (f \sqcap -q)^+ \sqcup (f \sqcap q)^{T\star} * (f \sqcap -q)^+ \sqcup (f \sqcap q)^{T+}$
using *comp-associative mult-left-dist-sup mult-right-dist-sup sup-assoc sup-commute* **by** *simp*
also have ... $= (f \sqcap -q)^\star * e * (f \sqcap q)^{T\star} * (f \sqcap -q)^\star \sqcup (f \sqcap q)^{T\star} * (f \sqcap -q)^+ \sqcup (f \sqcap q)^{T+}$
by (*metis star.circ-back-loop-fixpoint star-plus sup-monoid.add-commute mult-assoc*)
also have ... $\le f^\star * e * f^{T\star} * (f \sqcap -q)^\star \sqcup (f \sqcap q)^{T\star} * (f \sqcap -q)^+ \sqcup (f \sqcap q)^{T+}$
using *mult-left-isotone mult-right-isotone star-isotone sup-left-isotone conv-isotone order-trans inf-le1* **by** *meson*
also have ... $\le f^\star * e * f^{T\star} * f^\star \sqcup (f \sqcap q)^{T\star} * (f \sqcap -q)^+ \sqcup f^{T+}$

using *mult-left-isotone mult-right-isotone star-isotone sup-left-isotone sup-right-isotone conv-isotone order-trans inf-le1* **by** *meson*
also have ... $= f^\star * e * f^{T\star} * f^\star \sqcup (f \sqcap q)^{T+} * (f \sqcap -q)^+ \sqcup (f \sqcap -q)^+ \sqcup f^{T+}$
by (*simp add*: *star.circ-loop-fixpoint sup-monoid.add-assoc mult-assoc*)
also have ... $\leq f^\star * e * f^{T\star} * f^\star \sqcup (f \sqcap q)^{T+} * (f \sqcap -q)^+ \sqcup f^+ \sqcup f^{T+}$
using *mult-left-isotone mult-right-isotone star-isotone sup-left-isotone sup-right-isotone order-trans inf-le1* **by** *meson*
also have ... ≤ -1
using *7 8 9 assms*(*1*) **by** *simp*
finally show *?thesis*
by *simp*
qed

lemma *kruskal-exchange-acyclic-inv-1*:
assumes *acyclic f*
and *covector q*
shows *acyclic* $((f \sqcap -q) \sqcup (f \sqcap q)^T)$
using *kruskal-acyclic-inv*[**where** *e=bot*] **by** (*simp add*: *assms*)

lemma *kruskal-exchange-acyclic-inv-2*:
assumes *acyclic w*
and *injective w*
and $d \leq w$
and *bijective* $(d^T * top)$
and *bijective* $(e * top)$
and $d \leq top * e^T * w^{T\star}$
and $w * e^T * top = bot$
shows *acyclic* $((w \sqcap -d) \sqcup e)$
proof −
let $?v = w \sqcap -d$
let $?w = ?v \sqcup e$
have $d^T * top \leq w^\star * e * top$
by (*metis assms*(*6*) *comp-associative comp-inf.star.circ-decompose-9 comp-inf.star-star-absorb comp-isotone conv-dist-comp conv-involutive conv-order conv-star-commute conv-top inf.cobounded1 vector-top-closed*)
hence *1*: $e * top \leq w^{T\star} * d^T * top$
by (*metis assms*(*4*,*5*) *bijective-reverse comp-associative conv-star-commute*)
have *2*: $?v * d^T * top = bot$
by (*simp add*: *assms*(*2*,*3*) *kruskal-exchange-acyclic-inv-3*)
have $?v * w^{T+} * d^T * top \leq w * w^{T+} * d^T * top$
by (*simp add*: *mult-left-isotone*)
also have ... $\leq w^{T\star} * d^T * top$
by (*metis assms*(*2*) *mult-left-isotone mult-1-left mult-assoc*)
finally have $?v * w^{T\star} * d^T * top \leq w^{T\star} * d^T * top$
using *2* **by** (*metis bot-least comp-associative mult-right-dist-sup star.circ-back-loop-fixpoint star.circ-plus-same sup-least*)
hence *3*: $?v^\star * e * top \leq w^{T\star} * d^T * top$
using *1* **by** (*simp add*: *comp-associative star-left-induct sup-least*)
have $d * e^T \leq bot$

by (*metis assms*(*3*,*7*) *conv-bot conv-dist-comp conv-involutive conv-top order.trans inf.absorb2 inf.cobounded2 inf-commute le-bot p-antitone-iff p-top schroeder-4-p top-left-mult-increasing*)
hence *4*: $e^T * top \le -(d^T * top)$
by (*metis* (*no-types*) *comp-associative inf.cobounded2 le-bot p-antitone-iff schroeder-3-p semiring.mult-zero-left*)
have $?v^T * -(d^T * top) \le -(d^T * top)$
using *schroeder-3-p mult-assoc 2* **by** *simp*
hence $?v^{T\star} * e^T * top \le -(d^T * top)$
using *4* **by** (*simp add*: *comp-associative star-left-induct sup-least*)
hence *5*: $d^T * top \le -(?v^{T\star} * e^T * top)$
by (*simp add*: *p-antitone-iff*)
have $w * ?v^{T\star} * e^T * top = w * e^T * top \sqcup w * ?v^{T+} * e^T * top$
by (*metis star-left-unfold-equal mult-right-dist-sup mult-left-dist-sup mult-1-right mult-assoc*)
also have ... $= w * ?v^{T+} * e^T * top$
using *assms*(*7*) **by** *simp*
also have ... $\le w * w^T * ?v^{T\star} * e^T * top$
by (*simp add*: *comp-associative conv-isotone mult-left-isotone mult-right-isotone*)
also have ... $\le ?v^{T\star} * e^T * top$
by (*metis assms*(*2*) *mult-1-left mult-left-isotone*)
finally have $w * ?v^{T\star} * e^T * top \le --(?v^{T\star} * e^T * top)$
by (*simp add*: *p-antitone p-antitone-iff*)
hence $w^T * -(?v^{T\star} * e^T * top) \le -(?v^{T\star} * e^T * top)$
using *comp-associative schroeder-3-p* **by** *simp*
hence *6*: $w^{T\star} * d^T * top \le -(?v^{T\star} * e^T * top)$
using *5* **by** (*simp add*: *comp-associative star-left-induct sup-least*)
have $e * ?v^\star * e \le e * ?v^\star * e * top$
by (*simp add*: *top-right-mult-increasing*)
also have ... $\le e * w^{T\star} * d^T * top$
using *3* **by** (*simp add*: *comp-associative mult-right-isotone*)
also have ... $\le e * -(?v^{T\star} * e^T * top)$
using *6* **by** (*simp add*: *comp-associative mult-right-isotone*)
also have ... $\le bot$
by (*metis conv-complement-sub-leq conv-dist-comp conv-involutive conv-star-commute le-bot mult-right-sub-dist-sup-right p-bot regular-closed-bot star.circ-back-loop-fixpoint*)
finally have *7*: $e * ?v^\star * e = bot$
by (*simp add*: *order.antisym*)
hence $?v^\star * e \le -1$
by (*metis bot-least comp-associative comp-commute-below-diversity ex231d order-lesseq-imp semiring.mult-zero-left star.circ-left-top*)
hence *8*: $?v^\star * e * ?v^\star \le -1$
by (*metis comp-associative comp-commute-below-diversity star.circ-transitive-equal*)
have $1 \sqcap ?w^+ = 1 \sqcap ?w * ?v^\star * (e * ?v^\star)^\star$
by (*simp add*: *star-sup-1 mult-assoc*)
also have ... $= 1 \sqcap ?w * ?v^\star * (e * ?v^\star \sqcup 1)$

using *7* **by** (*metis star.circ-mult-1 star-absorb sup-monoid.add-commute mult-assoc*)
also have ... $= 1 \sqcap (?v^+ * e * ?v^\star \sqcup ?v^+ \sqcup e * ?v^\star * e * ?v^\star \sqcup e * ?v^\star)$
by (*simp add*: *comp-associative mult-left-dist-sup mult-right-dist-sup sup-assoc sup-commute sup-left-commute*)
also have ... $= 1 \sqcap (?v^+ * e * ?v^\star \sqcup ?v^+ \sqcup e * ?v^\star)$
using *7* **by** *simp*
also have ... $= 1 \sqcap (?v^\star * e * ?v^\star \sqcup ?v^+)$
by (*metis* (*mono-tags*, *opaque-lifting*) *comp-associative star.circ-loop-fixpoint sup-assoc sup-commute*)
also have ... $\leq 1 \sqcap (?v^\star * e * ?v^\star \sqcup w^+)$
using *comp-inf.mult-right-isotone comp-isotone semiring.add-right-mono star-isotone sup-commute* **by** *simp*
also have ... $= (1 \sqcap ?v^\star * e * ?v^\star) \sqcup (1 \sqcap w^+)$
by (*simp add*: *inf-sup-distrib1*)
also have ... $= 1 \sqcap ?v^\star * e * ?v^\star$
by (*metis assms*(*1*) *inf-commute pseudo-complement sup-bot-right*)
also have ... $=$ *bot*
using *8 p-antitone-iff pseudo-complement* **by** *simp*
finally show *?thesis*
using *le-bot p-antitone-iff pseudo-complement* **by** *auto*
qed

4.2.2 Exchange gives Spanning Trees

The lemmas in this section are used to show that the relation after exchange represents a spanning tree.

lemma *inf-star-import*:
assumes $x \leq z$
and *univalent z*
and *reflexive y*
and *regular z*
shows $x^\star * y \sqcap z^\star \leq x^\star * (y \sqcap z^\star)$
proof –
have *1*: $y \leq x^\star * (y \sqcap z^\star) \sqcup -z^\star$
by (*metis assms*(*4*) *pp-dist-star shunting-var-p star.circ-loop-fixpoint sup.cobounded2*)
have $x * -z^\star \sqcap z^+ \leq x * (-z^\star \sqcap x^T * z^+)$
by (*simp add*: *dedekind-1*)
also have ... $\leq x * (-z^\star \sqcap z^T * z^+)$
using *assms*(*1*) *comp-inf.mult-right-isotone conv-isotone mult-left-isotone mult-right-isotone* **by** *simp*
also have ... $\leq x * (-z^\star \sqcap 1 * z^\star)$
by (*metis assms*(*2*) *comp-associative comp-inf.mult-right-isotone mult-left-isotone mult-right-isotone*)
finally have *2*: $x * -z^\star \sqcap z^+ =$ *bot*
by (*simp add*: *order.antisym*)
have $x * -z^\star \sqcap z^\star = (x * -z^\star \sqcap z^+) \sqcup (x * -z^\star \sqcap 1)$
by (*metis comp-inf.semiring.distrib-left star-left-unfold-equal sup-commute*)

also have ... $\leq x^\star * (y \sqcap z^\star)$
using *2* **by** (*simp add*: *assms*(*3*) *inf.coboundedI2 reflexive-mult-closed star.circ-reflexive*)
finally have $x * -z^\star \leq x^\star * (y \sqcap z^\star) \sqcup -z^\star$
by (*metis assms*(*4*) *pp-dist-star shunting-var-p*)
hence $x * (x^\star * (y \sqcap z^\star) \sqcup -z^\star) \leq x^\star * (y \sqcap z^\star) \sqcup -z^\star$
by (*metis le-supE le-supI mult-left-dist-sup star.circ-loop-fixpoint sup.cobounded1*)
hence $x^\star * y \leq x^\star * (y \sqcap z^\star) \sqcup -z^\star$
using *1* **by** (*simp add*: *star-left-induct*)
hence $x^\star * y \sqcap --z^\star \leq x^\star * (y \sqcap z^\star)$
using *shunting-var-p* **by** *simp*
thus *?thesis*
using *order.trans inf.sup-right-isotone pp-increasing* **by** *blast*
qed

lemma *kruskal-exchange-forest-components-inv*:
assumes *injective* $((w \sqcap -d) \sqcup e)$
and *regular* d
and $e * top * e = e$
and $d \leq top * e^T * w^{T\star}$
and $w * e^T * top = bot$
and *injective* w
and $d \leq w$
and $d \leq (w \sqcap -d)^{T\star} * e^T * top$
shows *forest-components* $w \leq$ *forest-components* $((w \sqcap -d) \sqcup e)$
proof −
let $?v = w \sqcap -d$
let $?w = ?v \sqcup e$
let $?f =$ *forest-components* $?w$
have *1*: $?v * d^T * top = bot$
by (*simp add*: *assms*(*6*,*7*) *kruskal-exchange-acyclic-inv-3*)
have *2*: $d * e^T \leq bot$
by (*metis assms*(*5*,*7*) *conv-bot conv-dist-comp conv-involutive conv-top order.trans inf.absorb2 inf.cobounded2 inf-commute le-bot p-antitone-iff p-top schroeder-4-p top-left-mult-increasing*)
have $w^\star * e^T * top = e^T * top$
by (*metis assms*(*5*) *conv-bot conv-dist-comp conv-involutive conv-star-commute star.circ-top star-absorb*)
hence $w^\star * e^T * top \leq -(d^T * top)$
using *2* **by** (*metis* (*no-types*) *comp-associative inf.cobounded2 le-bot p-antitone-iff schroeder-3-p semiring.mult-zero-left*)
hence *3*: $e^T * top \leq -(w^{T\star} * d^T * top)$
by (*metis conv-star-commute p-antitone-iff schroeder-3-p mult-assoc*)
have $?v * w^{T\star} * d^T * top = ?v * d^T * top \sqcup ?v * w^{T+} * d^T * top$
by (*metis comp-associative mult-left-dist-sup star.circ-loop-fixpoint sup-commute*)
also have ... $\leq w * w^{T+} * d^T * top$
using *1* **by** (*simp add*: *mult-left-isotone*)

also have ... $\leq w^{T\star} * d^T * top$
by (*metis assms*(*6*) *mult-assoc mult-1-left mult-left-isotone*)
finally have $?v * w^{T\star} * d^T * top \leq --(w^{T\star} * d^T * top)$
using *p-antitone p-antitone-iff* **by** *auto*
hence *4*: $?v^T * -(w^{T\star} * d^T * top) \leq -(w^{T\star} * d^T * top)$
using *comp-associative schroeder-3-p* **by** *simp*
have *5*: *injective ?v*
using *assms*(*1*) *conv-dist-sup mult-left-dist-sup mult-right-dist-sup* **by** *simp*
have $?v * ?v^{T\star} * e^T * top = ?v * e^T * top \sqcup ?v * ?v^{T+} * e^T * top$
by (*metis comp-associative mult-left-dist-sup star.circ-loop-fixpoint sup-commute*)
also have ... $\leq w * e^T * top \sqcup ?v * ?v^{T+} * e^T * top$
using *mult-left-isotone sup-left-isotone* **by** *simp*
also have ... $\leq w * e^T * top \sqcup ?v^{T\star} * e^T * top$
using *5* **by** (*metis mult-assoc mult-1-left mult-left-isotone sup-right-isotone*)
finally have $?v * ?v^{T\star} * e^T * top \leq ?v^{T\star} * e^T * top$
by (*simp add*: *assms*(*5*))
hence $?v^\star * d * top \leq ?v^{T\star} * e^T * top$
by (*metis assms*(*8*) *star-left-induct sup-least comp-associative mult-right-sub-dist-sup-right sup.orderE vector-top-closed*)
also have ... $\leq -(w^{T\star} * d^T * top)$
using *3 4* **by** (*simp add*: *comp-associative star-left-induct*)
also have ... $\leq -(d^T * top)$
by (*metis p-antitone star.circ-left-top star-outer-increasing mult-assoc*)
finally have *6*: $?v^\star * d * top \leq -(d^T * top)$
by *simp*
have $d^T * top \leq w^\star * e * top$
by (*metis assms*(*4*) *comp-associative comp-inf.star.circ-sup-2 comp-isotone conv-dist-comp conv-involutive conv-order conv-star-commute conv-top vector-top-closed*)
also have ... $\leq (?v \sqcup d)^\star * e * top$
by (*metis assms*(*2*) *comp-inf.semiring.distrib-left maddux-3-11-pp mult-left-isotone star-isotone sup.cobounded2 sup-commute sup-inf-distrib1*)
also have ... $= ?v^\star * (d * ?v^\star)^\star * e * top$
by (*simp add*: *star-sup-1*)
also have ... $= ?v^\star * e * top \sqcup ?v^\star * d * ?v^\star * (d * ?v^\star)^\star * e * top$
by (*metis semiring.distrib-right star.circ-unfold-sum star-decompose-1 star-decompose-3 mult-assoc*)
also have ... $\leq ?v^\star * e * top \sqcup ?v^\star * d * top$
by (*metis comp-associative comp-isotone le-supI mult-left-dist-sup mult-right-dist-sup mult-right-isotone star.circ-decompose-5 star-decompose-3 sup.cobounded1 sup-commute top.extremum*)
finally have $d^T * top \leq ?v^\star * e * top \sqcup (d^T * top \sqcap ?v^\star * d * top)$
using *sup-inf-distrib2 sup-monoid.add-commute* **by** *simp*
hence $d^T * top \leq ?v^\star * e * top$
using *6* **by** (*metis inf-commute pseudo-complement sup-monoid.add-0-right*)
hence *7*: $d \leq top * e^T * ?v^{T\star}$
by (*metis comp-associative conv-dist-comp conv-involutive conv-isotone conv-star-commute conv-top order.trans top-right-mult-increasing*)

have *8*: $?v \leq ?f$
using *forest-components-increasing le-supE* **by** *blast*
have $d \leq ?v^{T\star} * e^T * top \sqcap top * e^T * ?v^{T\star}$
using *7* *assms*(*8*) **by** *simp*
also have ... $= ?v^{T\star} * e^T * top * e^T * ?v^{T\star}$
by (*metis inf-top-right vector-inf-comp vector-top-closed mult-assoc*)
also have ... $= ?v^{T\star} * e^T * ?v^{T\star}$
by (*metis assms*(*3*) *comp-associative conv-dist-comp conv-top*)
also have ... $\leq ?v^{T\star} * e^T * ?f$
using *8* **by** (*metis assms*(*1*) *forest-components-equivalence cancel-separate-1 conv-dist-comp conv-order mult-left-isotone star-involutive star-isotone*)
also have ... $\leq ?v^{T\star} * ?f * ?f$
by (*metis assms*(*1*) *forest-components-equivalence forest-components-increasing conv-isotone le-supE mult-left-isotone mult-right-isotone*)
also have ... $\leq ?f * ?f * ?f$
by (*metis comp-associative comp-isotone conv-dist-sup star.circ-loop-fixpoint star-isotone sup.cobounded1 sup.cobounded2*)
also have ... $= ?f$
by (*simp add*: *assms*(*1*) *forest-components-equivalence preorder-idempotent*)
finally have $w \leq ?f$
using *8* **by** (*metis assms*(*2*) *shunting-var-p sup.orderE*)
thus *?thesis*
using *assms*(*1*) *forest-components-idempotent forest-components-isotone* **by** *fastforce*
qed

lemma *kruskal-spanning-inv*:
assumes *injective* $((f \sqcap -q) \sqcup (f \sqcap q)^T \sqcup e)$
and *regular* q
and *regular* e
and $(-h \sqcap --g)^\star \leq$ *forest-components* f
shows *components* $(-(h \sqcap -e \sqcap -e^T) \sqcap g) \leq$ *forest-components* $((f \sqcap -q) \sqcup (f \sqcap q)^T \sqcup e)$
proof $-$
let $?f = (f \sqcap -q) \sqcup (f \sqcap q)^T \sqcup e$
let $?h = h \sqcap -e \sqcap -e^T$
let $?F =$ *forest-components* f
let $?FF =$ *forest-components* $?f$
have *1*: *equivalence* $?FF$
using *assms*(*1*) *forest-components-equivalence* **by** *simp*
hence *2*: $?f * ?FF \leq ?FF$
using *order.trans forest-components-increasing mult-left-isotone* **by** *blast*
have *3*: $?f^T * ?FF \leq ?FF$
using *1* **by** (*metis forest-components-increasing mult-left-isotone conv-isotone preorder-idempotent*)
have $(f \sqcap q) * ?FF \leq ?f^T * ?FF$
using *conv-dist-sup conv-involutive sup-assoc sup-left-commute mult-left-isotone* **by** *simp*
hence *4*: $(f \sqcap q) * ?FF \leq ?FF$

using *3 order.trans* **by** *blast*
have $(f \sqcap -q) * ?FF \leq ?f * ?FF$
using *le-supI1 mult-left-isotone* **by** *simp*
hence $(f \sqcap -q) * ?FF \leq ?FF$
using *2 order.trans* **by** *blast*
hence $((f \sqcap q) \sqcup (f \sqcap -q)) * ?FF \leq ?FF$
using *4 mult-right-dist-sup* **by** *simp*
hence $f * ?FF \leq ?FF$
by (*metis assms*(*2*) *maddux-3-11-pp*)
hence *5*: $f^\star * ?FF \leq ?FF$
using *star-left-induct-mult-iff* **by** *simp*
have $(f \sqcap -q)^T * ?FF \leq ?f^T * ?FF$
by (*meson conv-isotone order.trans mult-left-isotone sup.cobounded1*)
hence *6*: $(f \sqcap -q)^T * ?FF \leq ?FF$
using *3 order.trans* **by** *blast*
have $(f \sqcap q)^T * ?FF \leq ?f * ?FF$
by (*simp add*: *mult-left-isotone sup.left-commute sup-assoc*)
hence $(f \sqcap q)^T * ?FF \leq ?FF$
using *2 order.trans* **by** *blast*
hence $((f \sqcap -q)^T \sqcup (f \sqcap q)^T) * ?FF \leq ?FF$
using *6 mult-right-dist-sup* **by** *simp*
hence $f^T * ?FF \leq ?FF$
by (*metis assms*(*2*) *conv-dist-sup maddux-3-11-pp*)
hence *7*: $?F * ?FF \leq ?FF$
using *5 star-left-induct mult-assoc* **by** *simp*
have *8*: $e * ?FF \leq ?FF$
using *2* **by** (*simp add*: *mult-right-dist-sup mult-left-isotone*)
have $e^T * ?FF \leq ?f^T * ?FF$
by (*simp add*: *mult-left-isotone conv-isotone*)
also have ... $\leq ?FF * ?FF$
using *1* **by** (*metis forest-components-increasing mult-left-isotone conv-isotone*)
finally have $e^T * ?FF \leq ?FF$
using *1 preorder-idempotent* **by** *auto*
hence *9*: $(?F \sqcup e \sqcup e^T) * ?FF \leq ?FF$
using *7 8 mult-right-dist-sup* **by** *simp*
have *components* $(-?h \sqcap g) \leq ((-h \sqcap --g) \sqcup e \sqcup e^T)^\star$
by (*metis assms*(*3*) *comp-inf.mult-left-sub-dist-sup-left conv-complement p-dist-inf pp-dist-inf regular-closed-p star-isotone sup-inf-distrib2 sup-monoid.add-assoc*)
also have ... $\leq ((-h \sqcap --g)^\star \sqcup e \sqcup e^T)^\star$
using *star.circ-increasing star-isotone sup-left-isotone* **by** *simp*
also have ... $\leq (?F \sqcup e \sqcup e^T)^\star$
using *assms*(*4*) *sup-left-isotone star-isotone* **by** *simp*
also have ... $\leq ?FF$
using *1 9 star-left-induct* **by** *force*
finally show *?thesis*
by *simp*
qed

lemma *kruskal-exchange-spanning-inv-1*:
  assumes *injective* $((w \sqcap -q) \sqcup (w \sqcap q)^T)$
    and *regular* $(w \sqcap q)$
    and *components* $g \leq$ *forest-components* w
  shows *components* $g \leq$ *forest-components* $((w \sqcap -q) \sqcup (w \sqcap q)^T)$
proof −
  let $?p = w \sqcap q$
  let $?w = (w \sqcap -q) \sqcup ?p^T$
  have *1*: $w \sqcap -?p \leq$ *forest-components* $?w$
    by (*metis forest-components-increasing inf-import-p le-supE*)
  have $w \sqcap ?p \leq ?w^T$
    by (*simp add*: *conv-dist-sup*)
  also have ... $\leq$ *forest-components* $?w$
    by (*metis assms*(*1*) *conv-isotone forest-components-equivalence*
forest-components-increasing)
  finally have $w \sqcap (?p \sqcup -?p) \leq$ *forest-components* $?w$
    using *1 inf-sup-distrib1* **by** *simp*
  hence $w \leq$ *forest-components* $?w$
    by (*metis assms*(*2*) *inf-top-right stone*)
  hence *2*: $w^\star \leq$ *forest-components* $?w$
    using *assms*(*1*) *star-isotone forest-components-star* **by** *force*
  hence *3*: $w^{T\star} \leq$ *forest-components* $?w$
    using *assms*(*1*) *conv-isotone conv-star-commute forest-components-equivalence*
by *force*
  have *components* $g \leq$ *forest-components* w
    using *assms*(*3*) **by** *simp*
  also have ... $\leq$ *forest-components* $?w *$ *forest-components* $?w$
    using *2 3 mult-isotone* **by** *simp*
  also have ... $=$ *forest-components* $?w$
    using *assms*(*1*) *forest-components-equivalence preorder-idempotent* **by** *simp*
  finally show *?thesis*
    by *simp*
qed

lemma *kruskal-exchange-spanning-inv-2*:
  assumes *injective* w
    and $w^\star * e^T = e^T$
    and $f \sqcup f^T \leq (w \sqcap -d \sqcap -d^T) \sqcup (w^T \sqcap -d \sqcap -d^T)$
    and $d \leq$ *forest-components* $f * e^T * top$
  shows $d \leq (w \sqcap -d)^{T\star} * e^T * top$
proof −
  have *1*: $(w \sqcap -d \sqcap -d^T) * (w^T \sqcap -d \sqcap -d^T) \leq 1$
    using *assms*(*1*) *comp-isotone order.trans inf.cobounded1* **by** *blast*
  have $d \leq$ *forest-components* $f * e^T * top$
    using *assms*(*4*) **by** *simp*
  also have ... $\leq (f \sqcup f^T)^\star * (f \sqcup f^T)^\star * e^T * top$
    by (*simp add*: *comp-isotone star-isotone*)
  also have ... $= (f \sqcup f^T)^\star * e^T * top$
    by (*simp add*: *star.circ-transitive-equal*)

also have ... $\leq ((w \sqcap -d \sqcap -d^T) \sqcup (w^T \sqcap -d \sqcap -d^T))^\star * e^T * top$
  using *assms*(*3*) **by** (*simp add*: *comp-isotone star-isotone*)
also have ... $= (w^T \sqcap -d \sqcap -d^T)^\star * (w \sqcap -d \sqcap -d^T)^\star * e^T * top$
  using *1 cancel-separate-1* **by** *simp*
also have ... $\leq (w^T \sqcap -d \sqcap -d^T)^\star * w^\star * e^T * top$
  by (*simp add*: *inf-assoc mult-left-isotone mult-right-isotone star-isotone*)
also have ... $= (w^T \sqcap -d \sqcap -d^T)^\star * e^T * top$
  using *assms*(*2*) *mult-assoc* **by** *simp*
also have ... $\leq (w^T \sqcap -d^T)^\star * e^T * top$
  using *mult-left-isotone conv-isotone star-isotone comp-inf.mult-right-isotone inf.cobounded2 inf.left-commute inf.sup-monoid.add-commute* **by** *presburger*
also have ... $= (w \sqcap -d)^{T\star} * e^T * top$
  using *conv-complement conv-dist-inf* **by** *presburger*
finally show *?thesis*
  by *simp*
qed

lemma *kruskal-spanning-inv-1*:
assumes $e \leq F$
  and *regular* e
  and *components* $(-h \sqcap g) \leq F$
  and *equivalence* F
shows *components* $(-(h \sqcap -e \sqcap -e^T) \sqcap g) \leq F$
proof −
have *1*: $F * F \leq F$
  using *assms*(*4*) **by** *simp*
hence *2*: $e * F \leq F$
  using *assms*(*1*) *mult-left-isotone order-lesseq-imp* **by** *blast*
have $e^T * F \leq F$
  by (*metis assms*(*1*,*4*) *conv-isotone mult-left-isotone preorder-idempotent*)
hence *3*: $(F \sqcup e \sqcup e^T) * F \leq F$
  using *1 2 mult-right-dist-sup* **by** *simp*
have *components* $(-(h \sqcap -e \sqcap -e^T) \sqcap g) \leq ((-h \sqcap --g) \sqcup e \sqcup e^T)^\star$
  by (*metis assms*(*2*) *comp-inf.mult-left-sub-dist-sup-left conv-complement p-dist-inf pp-dist-inf regular-closed-p star-isotone sup-inf-distrib2 sup-monoid.add-assoc*)
also have ... $\leq ((-h \sqcap --g)^\star \sqcup e \sqcup e^T)^\star$
  using *sup-left-isotone star.circ-increasing star-isotone* **by** *simp*
also have ... $\leq (F \sqcup e \sqcup e^T)^\star$
  using *assms*(*3*) *sup-left-isotone star-isotone* **by** *simp*
also have ... $\leq F$
  using *3 assms*(*4*) *star-left-induct* **by** *force*
finally show *?thesis*
  by *simp*
qed

lemma *kruskal-reroot-edge*:
assumes *injective* $(e^T * top)$
  and *acyclic* w

shows $((w \sqcap -(top * e * w^{T\star})) \sqcup (w \sqcap top * e * w^{T\star})^T) * e^T = bot$
proof $-$
let $?q = top * e * w^{T\star}$
let $?p = w \sqcap ?q$
let $?w = (w \sqcap -?q) \sqcup ?p^T$
have $(w \sqcap -?q) * e^T * top = w * (e^T * top \sqcap -?q^T)$
by (*metis comp-associative comp-inf-vector-1 conv-complement covector-complement-closed vector-top-closed*)
also have ... $= w * (e^T * top \sqcap -(w^\star * e^T * top))$
by (*simp add*: *conv-dist-comp conv-star-commute mult-assoc*)
also have ... $= bot$
by (*metis comp-associative comp-inf.semiring.mult-not-zero inf.sup-relative-same-increasing inf-p mult-right-zero star.circ-loop-fixpoint sup-commute sup-left-divisibility*)
finally have *1*: $(w \sqcap -?q) * e^T * top = bot$
by *simp*
have $?p^T * e^T * top = (w^T \sqcap w^\star * e^T * top) * e^T * top$
by (*simp add*: *conv-dist-comp conv-star-commute mult-assoc conv-dist-inf*)
also have ... $= w^\star * e^T * top \sqcap w^T * e^T * top$
by (*simp add*: *inf-vector-comp vector-export-comp*)
also have ... $= (w^\star \sqcap w^T) * e^T * top$
using *assms*(*1*) *injective-comp-right-dist-inf mult-assoc* **by** *simp*
also have ... $= bot$
using *assms*(*2*) *acyclic-star-below-complement-1 semiring.mult-not-zero* **by** *blast*
finally have $?w * e^T * top = bot$
using *1 mult-right-dist-sup* **by** *simp*
thus *?thesis*
by (*metis star.circ-top star-absorb*)
qed

4.2.3 Exchange gives Minimum Spanning Trees

The lemmas in this section are used to show that the after exchange we obtain a minimum spanning tree. The following lemmas show that the relation characterising the edge across the cut is an arc.

lemma *kruskal-edge-arc*:
assumes *equivalence F*
and *forest w*
and *arc e*
and *regular F*
and $F \leq$ *forest-components* $(F \sqcap w)$
and *regular w*
and $w * e^T = bot$
and $e * F * e = bot$
and $e^T \leq w^\star$
shows *arc* $(w \sqcap top * e^T * w^{T\star} \sqcap F * e^T * top \sqcap top * e * -F)$
proof (*unfold arc-expanded*, *intro conjI*)
let $?E = top * e^T * w^{T\star}$

let *?F* $= F * e^T * top$
let *?G* $= top * e * -F$
let *?FF* $= F * e^T * e * F$
let *?GG* $= -F * e^T * e * -F$
let *?w* = *forest-components* $(F \sqcap w)$
have $F \sqcap w^{T\star} \le$ *forest-components* $(F \sqcap w) \sqcap w^{T\star}$
  by (*simp add*: *assms*(*5*) *inf.coboundedI1*)
also have ... $\le (F \sqcap w)^{T\star} * ((F \sqcap w)^\star \sqcap w^{T\star})$
  apply (*rule inf-star-import*)
  apply (*simp add*: *conv-isotone*)
  apply (*simp add*: *assms*(*2*))
  apply (*simp add*: *star.circ-reflexive*)
  by (*metis assms*(*6*) *conv-complement*)
also have ... $\le (F \sqcap w)^{T\star} * (w^\star \sqcap w^{T\star})$
  using *comp-inf.mult-left-isotone mult-right-isotone star-isotone* **by** *simp*
also have ... $= (F \sqcap w)^{T\star}$
  by (*simp add*: *assms*(*2*) *acyclic-star-inf-conv*)
finally have $w * (F \sqcap w^{T\star}) * e^T * e \le w * (F \sqcap w)^{T\star} * e^T * e$
  by (*simp add*: *mult-left-isotone mult-right-isotone*)
also have ... $= w * e^T * e \sqcup w * (F \sqcap w)^{T+} * e^T * e$
  by (*metis comp-associative mult-left-dist-sup star.circ-loop-fixpoint*
sup-commute)
also have ... $= w * (F \sqcap w)^{T+} * e^T * e$
  by (*simp add*: *assms*(*7*))
also have ... $\le w * (F \sqcap w)^{T+}$
  by (*metis assms*(*3*) *arc-univalent mult-assoc mult-1-right mult-right-isotone*)
also have ... $\le w * w^T * (F \sqcap w)^{T\star}$
  by (*simp add*: *comp-associative conv-isotone mult-left-isotone*
mult-right-isotone)
also have ... $\le (F \sqcap w)^{T\star}$
  using *assms*(*2*) *coreflexive-comp-top-inf inf.sup-right-divisibility* **by** *auto*
also have ... $\le F^{T\star}$
  by (*simp add*: *conv-dist-inf star-isotone*)
finally have *1*: $w * (F \sqcap w^{T\star}) * e^T * e \le F$
  by (*metis assms*(*1*) *order.antisym mult-1-left mult-left-isotone*
star.circ-plus-same star.circ-reflexive star.left-plus-below-circ
star-left-induct-mult-iff)
have $F * e^T * e \le$ *forest-components* $(F \sqcap w) * e^T * e$
  by (*simp add*: *assms*(*5*) *mult-left-isotone*)
also have ... $\le$ *forest-components* $w * e^T * e$
  by (*simp add*: *comp-isotone conv-dist-inf star-isotone*)
also have ... $= w^{T\star} * e^T * e$
  by (*metis* (*no-types*) *assms*(*7*) *comp-associative conv-bot conv-dist-comp*
conv-involutive conv-star-commute star-absorb)
also have ... $\le w^{T\star}$
  by (*metis assms*(*3*) *arc-univalent mult-assoc mult-1-right mult-right-isotone*)
finally have *2*: $F * e^T * e \le w^{T\star}$
  by *simp*
have $w * F * e^T * e \le w * F * e^T * e * e^T * e$

using *comp-associative ex231c mult-right-isotone* **by** *simp*
also have ... $= w * (F * e^T * e \sqcap w^{T\star}) * e^T * e$
using *2* **by** (*simp add*: *comp-associative inf.absorb1*)
also have ... $\leq w * (F \sqcap w^{T\star}) * e^T * e$
by (*metis assms*(*3*) *arc-univalent mult-assoc mult-1-right mult-right-isotone mult-left-isotone inf.sup-left-isotone*)
also have ... $\leq F$
using *1* **by** *simp*
finally have *3*: $w * F * e^T * e \leq F$
by *simp*
hence $e^T * e * F * w^T \leq F$
by (*metis assms*(*1*) *conv-dist-comp conv-dist-inf conv-involutive inf.absorb-iff1 mult-assoc*)
hence $e^T * e * F * w^T \leq e^T * top \sqcap F$
by (*simp add*: *comp-associative mult-right-isotone*)
also have ... $\leq e^T * e * F$
by (*metis conv-involutive dedekind-1 inf-top-left mult-assoc*)
finally have *4*: $e^T * e * F * w^T \leq e^T * e * F$
by *simp*
have $(top * e)^T * (?F \sqcap w^{T\star}) = e^T * top * e * F * w^{T\star}$
by (*metis assms*(*1*) *comp-inf.star.circ-decompose-9 comp-inf.star-star-absorb conv-dist-comp conv-involutive conv-top covector-inf-comp-3 vector-top-closed mult-assoc*)
also have ... $= e^T * e * F * w^{T\star}$
by (*simp add*: *assms*(*3*) *arc-top-edge*)
also have ... $\leq e^T * e * F$
using *4* *star-right-induct-mult* **by** *simp*
also have ... $\leq F$
by (*metis assms*(*3*) *arc-injective conv-involutive mult-1-left mult-left-isotone*)
finally have *5*: $(top * e)^T * (?F \sqcap w^{T\star}) \leq F$
by *simp*
have $(?F \sqcap w) * w^{T+} = ?F \sqcap w * w^{T+}$
by (*simp add*: *vector-export-comp*)
also have ... $\leq ?F \sqcap w^{T\star}$
by (*metis assms*(*2*) *comp-associative inf.sup-right-isotone mult-left-isotone star.circ-transitive-equal star-left-unfold-equal sup.absorb-iff2 sup-monoid.add-assoc*)
also have *6*: ... $\leq top * e * F$
using *5* **by** (*metis assms*(*3*) *shunt-mapping conv-dist-comp conv-involutive conv-top*)
finally have *7*: $(?F \sqcap w) * w^{T+} \leq top * e * F$
by *simp*
have $e^T * top * e \leq 1$
by (*simp add*: *assms*(*3*) *point-injective*)
also have ... $\leq F$
by (*simp add*: *assms*(*1*))
finally have *8*: $e * -F * e^T \leq bot$
by (*metis p-antitone p-antitone-iff p-bot regular-closed-bot schroeder-3-p schroeder-4-p mult-assoc*)

have $?FF \sqcap w * (w^{T+} \sqcap ?GG) * w^T \leq ?F \sqcap w * (w^{T+} \sqcap ?GG) * w^T$
using *comp-inf.mult-left-isotone mult-isotone mult-assoc* **by** *simp*
also have ... $\leq ?F \sqcap w * (w^{T+} \sqcap ?G) * w^T$
by (*metis assms*(*3*) *arc-top-edge comp-inf.star.circ-decompose-9 comp-inf-covector inf.sup-right-isotone inf-le2 mult-left-isotone mult-right-isotone vector-top-closed mult-assoc*)
also have ... $= (?F \sqcap w) * (w^{T+} \sqcap ?G) * w^T$
by (*simp add*: *vector-export-comp*)
also have ... $= (?F \sqcap w) * w^{T+} * (?G^T \sqcap w^T)$
by (*simp add*: *covector-comp-inf covector-comp-inf-1 covector-mult-closed*)
also have ... $\leq top * e * F * (?G^T \sqcap w^T)$
using *7 mult-left-isotone* **by** *simp*
also have ... $\leq top * e * F * ?G^T$
by (*simp add*: *mult-right-isotone*)
also have ... $= top * e * -F * e^T * top$
by (*metis assms*(*1*) *conv-complement conv-dist-comp conv-top equivalence-comp-left-complement mult-assoc*)
finally have *9*: $?FF \sqcap w * (w^{T+} \sqcap ?GG) * w^T = bot$
using *8* **by** (*metis comp-associative covector-bot-closed le-bot vector-bot-closed*)
hence *10*: $?FF \sqcap w * (w^+ \sqcap ?GG) * w^T = bot$
using *assms*(*1*) *comp-associative conv-bot conv-complement conv-dist-comp conv-dist-inf conv-star-commute star.circ-plus-same* **by** *fastforce*
have $(w \sqcap ?E \sqcap ?F \sqcap ?G) * top * (w \sqcap ?E \sqcap ?F \sqcap ?G)^T = (?F \sqcap (w \sqcap ?E \sqcap ?G)) * top * ((w \sqcap ?E \sqcap ?G)^T \sqcap ?F^T)$
by (*simp add*: *conv-dist-inf inf-commute inf-left-commute*)
also have ... $= (?F \sqcap (w \sqcap ?E \sqcap ?G)) * top * (w \sqcap ?E \sqcap ?G)^T \sqcap ?F^T$
using *covector-comp-inf vector-conv-covector vector-mult-closed vector-top-closed* **by** *simp*
also have ... $= ?F \sqcap (w \sqcap ?E \sqcap ?G) * top * (w \sqcap ?E \sqcap ?G)^T \sqcap ?F^T$
by (*simp add*: *vector-export-comp*)
also have ... $= ?F \sqcap top * e * F \sqcap (w \sqcap ?E \sqcap ?G) * top * (w \sqcap ?E \sqcap ?G)^T$
by (*simp add*: *assms*(*1*) *conv-dist-comp inf-assoc inf-commute mult-assoc*)
also have ... $= ?F * e * F \sqcap (w \sqcap ?E \sqcap ?G) * top * (w \sqcap ?E \sqcap ?G)^T$
by (*metis comp-associative comp-inf-covector inf-top.left-neutral*)
also have ... $= ?FF \sqcap (w \sqcap ?E \sqcap ?G) * (top * (w \sqcap ?E \sqcap ?G)^T)$
using *assms*(*3*) *arc-top-edge comp-associative* **by** *simp*
also have ... $= ?FF \sqcap (w \sqcap ?E \sqcap ?G) * (top * (?G^T \sqcap (?E^T \sqcap w^T)))$
by (*simp add*: *conv-dist-inf inf-assoc inf-commute inf-left-commute*)
also have ... $= ?FF \sqcap (w \sqcap ?E \sqcap ?G) * (?G * (?E^T \sqcap w^T))$
by (*metis covector-comp-inf-1 covector-top-closed covector-mult-closed inf-top-left*)
also have ... $= ?FF \sqcap (w \sqcap ?E \sqcap ?G) * (?G \sqcap ?E) * w^T$
by (*metis covector-comp-inf-1 covector-top-closed mult-assoc*)
also have ... $= ?FF \sqcap (w \sqcap ?E) * (?G^T \sqcap ?G \sqcap ?E) * w^T$
by (*metis covector-comp-inf-1 covector-mult-closed inf.sup-monoid.add-assoc vector-top-closed*)
also have ... $= ?FF \sqcap w * (?E^T \sqcap ?G^T \sqcap ?G \sqcap ?E) * w^T$
by (*metis covector-comp-inf-1 covector-mult-closed inf.sup-monoid.add-assoc vector-top-closed*)

also have ... $=$ *?FF* $\sqcap$ $w * (?E^T \sqcap ?E \sqcap (?G^T \sqcap ?G)) * w^T$
by (*simp add*: *inf-commute inf-left-commute*)
also have ... $=$ *?FF* $\sqcap$ $w * (?E^T \sqcap ?E \sqcap (-F * e^T * top \sqcap ?G)) * w^T$
by (*simp add*: *assms*(*1*) *conv-complement conv-dist-comp mult-assoc*)
also have ... $=$ *?FF* $\sqcap$ $w * (?E^T \sqcap ?E \sqcap (-F * e^T * ?G)) * w^T$
by (*metis comp-associative comp-inf-covector inf-top.left-neutral*)
also have ... $=$ *?FF* $\sqcap$ $w * (?E^T \sqcap ?E \sqcap ?GG) * w^T$
by (*metis assms*(*3*) *arc-top-edge comp-associative*)
also have ... $=$ *?FF* $\sqcap$ $w * (w^\star * e * top \sqcap ?E \sqcap ?GG) * w^T$
by (*simp add*: *comp-associative conv-dist-comp conv-star-commute*)
also have ... $=$ *?FF* $\sqcap$ $w * (w^\star * e * ?E \sqcap ?GG) * w^T$
by (*metis comp-associative comp-inf-covector inf-top.left-neutral*)
also have ... $\leq$ *?FF* $\sqcap$ $w * (w^\star * w^{T\star} \sqcap ?GG) * w^T$
by (*metis assms*(*3*) *mult-assoc mult-1-right mult-left-isotone mult-right-isotone inf.sup-left-isotone inf.sup-right-isotone arc-expanded*)
also have ... $=$ *?FF* $\sqcap$ $w * ((w^+ \sqcup 1 \sqcup w^{T\star}) \sqcap ?GG) * w^T$
by (*simp add*: *assms*(*2*) *cancel-separate-eq star-left-unfold-equal sup-monoid.add-commute*)
also have ... $=$ *?FF* $\sqcap$ $w * ((w^+ \sqcup 1 \sqcup w^{T+}) \sqcap ?GG) * w^T$
using *star.circ-plus-one star-left-unfold-equal sup-assoc* **by** *presburger*
also have ... $=$ (*?FF* $\sqcap$ $w * (w^+ \sqcap ?GG) * w^T) \sqcup$ (*?FF* $\sqcap$ $w * (1 \sqcap ?GG) * w^T) \sqcup$ (*?FF* $\sqcap$ $w * (w^{T+} \sqcap ?GG) * w^T$)
by (*simp add*: *inf-sup-distrib1 inf-sup-distrib2 semiring.distrib-left semiring.distrib-right*)
also have ... $\leq w * (1 \sqcap ?GG) * w^T$
using *9 10* **by** *simp*
also have ... $\leq w * w^T$
by (*metis inf.cobounded1 mult-1-right mult-left-isotone mult-right-isotone*)
also have ... $\leq$ *1*
by (*simp add*: *assms*(*2*))
finally show $(w \sqcap ?E \sqcap ?F \sqcap ?G) * top * (w \sqcap ?E \sqcap ?F \sqcap ?G)^T \leq 1$
by *simp*
have $w^{T+} \sqcap -F * e^T * e * -F \sqcap w^T * F * e^T * e * F * w \leq w^{T+} \sqcap ?G \sqcap w^T * F * e^T * e * F * w$
using *top-greatest inf.sup-left-isotone inf.sup-right-isotone mult-left-isotone* **by** *simp*
also have ... $\leq w^{T+} \sqcap ?G \sqcap w^T * ?F$
using *comp-associative inf.sup-right-isotone mult-right-isotone top.extremum* **by** *presburger*
also have ... $= w^T * (w^{T\star} \sqcap ?F) \sqcap ?G$
using *assms*(*2*) *inf-assoc inf-commute inf-left-commute univalent-comp-left-dist-inf* **by** *simp*
also have ... $\leq w^T * (top * e * F) \sqcap ?G$
using *6* **by** (*metis inf.sup-monoid.add-commute inf.sup-right-isotone mult-right-isotone*)
also have ... $\leq top * e * F \sqcap ?G$
by (*metis comp-associative comp-inf-covector mult-left-isotone top.extremum*)
also have ... $=$ *bot*
by (*metis assms*(*3*) *conv-dist-comp conv-involutive conv-top inf-p*

mult-right-zero univalent-comp-left-dist-inf)
finally have *11*: $w^{T+} \sqcap -F * e^T * e * -F \sqcap w^T * F * e^T * e * F * w = bot$
by (*simp add*: *order.antisym*)
hence *12*: $w^+ \sqcap -F * e^T * e * -F \sqcap w^T * F * e^T * e * F * w = bot$
using *assms*(*1*) *comp-associative conv-bot conv-complement conv-dist-comp conv-dist-inf conv-star-commute star.circ-plus-same* **by** *fastforce*
have $(w \sqcap ?E \sqcap ?F \sqcap ?G)^T * top * (w \sqcap ?E \sqcap ?F \sqcap ?G) = ((w \sqcap ?E \sqcap ?G)^T \sqcap ?F^T) * top * (?F \sqcap (w \sqcap ?E \sqcap ?G))$
by (*simp add*: *conv-dist-inf inf-commute inf-left-commute*)
also have ... $= (w \sqcap ?E \sqcap ?G)^T * ?F * (?F \sqcap (w \sqcap ?E \sqcap ?G))$
by (*simp add*: *covector-inf-comp-3 vector-mult-closed*)
also have ... $= (w \sqcap ?E \sqcap ?G)^T * (?F \sqcap ?F^T) * (w \sqcap ?E \sqcap ?G)$
using *covector-comp-inf covector-inf-comp-3 vector-conv-covector vector-mult-closed* **by** *simp*
also have ... $= (w \sqcap ?E \sqcap ?G)^T * (?F \sqcap ?F^T) * (w \sqcap ?E) \sqcap ?G$
by (*simp add*: *comp-associative comp-inf-covector*)
also have ... $= (w \sqcap ?E \sqcap ?G)^T * (?F \sqcap ?F^T) * w \sqcap ?E \sqcap ?G$
by (*simp add*: *comp-associative comp-inf-covector*)
also have ... $= (?G^T \sqcap (?E^T \sqcap w^T)) * (?F \sqcap ?F^T) * w \sqcap ?E \sqcap ?G$
by (*simp add*: *conv-dist-inf inf.left-commute inf.sup-monoid.add-commute*)
also have ... $= ?G^T \sqcap (?E^T \sqcap w^T) * (?F \sqcap ?F^T) * w \sqcap ?E \sqcap ?G$
by (*metis* (*no-types*) *comp-associative conv-dist-comp conv-top vector-export-comp*)
also have ... $= ?G^T \sqcap ?E^T \sqcap w^T * (?F \sqcap ?F^T) * w \sqcap ?E \sqcap ?G$
by (*metis* (*no-types*) *comp-associative inf-assoc conv-dist-comp conv-top vector-export-comp*)
also have ... $= ?E^T \sqcap ?E \sqcap (?G^T \sqcap ?G) \sqcap w^T * (?F \sqcap ?F^T) * w$
by (*simp add*: *inf-assoc inf.left-commute inf.sup-monoid.add-commute*)
also have ... $= w^\star * e * top \sqcap ?E \sqcap (?G^T \sqcap ?G) \sqcap w^T * (?F \sqcap ?F^T) * w$
by (*simp add*: *comp-associative conv-dist-comp conv-star-commute*)
also have ... $= w^\star * e * ?E \sqcap (?G^T \sqcap ?G) \sqcap w^T * (?F \sqcap ?F^T) * w$
by (*metis comp-associative comp-inf-covector inf-top.left-neutral*)
also have ... $\le w^\star * w^{T\star} \sqcap (?G^T \sqcap ?G) \sqcap w^T * (?F \sqcap ?F^T) * w$
by (*metis assms*(*3*) *mult-assoc mult-1-right mult-left-isotone mult-right-isotone inf.sup-left-isotone arc-expanded*)
also have ... $= w^\star * w^{T\star} \sqcap (-F * e^T * top \sqcap ?G) \sqcap w^T * (?F \sqcap ?F^T) * w$
by (*simp add*: *assms*(*1*) *conv-complement conv-dist-comp mult-assoc*)
also have ... $= w^\star * w^{T\star} \sqcap -F * e^T * ?G \sqcap w^T * (?F \sqcap ?F^T) * w$
by (*metis comp-associative comp-inf-covector inf-top.left-neutral*)
also have ... $= w^\star * w^{T\star} \sqcap -F * e^T * e * -F \sqcap w^T * (?F \sqcap ?F^T) * w$
by (*metis assms*(*3*) *arc-top-edge mult-assoc*)
also have ... $= w^\star * w^{T\star} \sqcap -F * e^T * e * -F \sqcap w^T * (?F \sqcap top * e * F) * w$
by (*simp add*: *assms*(*1*) *conv-dist-comp mult-assoc*)
also have ... $= w^\star * w^{T\star} \sqcap -F * e^T * e * -F \sqcap w^T * (?F * e * F) * w$
by (*metis comp-associative comp-inf-covector inf-top.left-neutral*)
also have ... $= w^\star * w^{T\star} \sqcap -F * e^T * e * -F \sqcap w^T * F * e^T * e * F * w$
by (*metis assms*(*3*) *arc-top-edge mult-assoc*)
also have ... $= (w^+ \sqcup 1 \sqcup w^{T\star}) \sqcap -F * e^T * e * -F \sqcap w^T * F * e^T * e * F * w$

by (*simp add*: *assms*(*2*) *cancel-separate-eq star-left-unfold-equal sup-monoid.add-commute*)
also have ... $= (w^+ \sqcup 1 \sqcup w^{T+}) \sqcap -F * e^T * e * -F \sqcap w^T * F * e^T * e * F * w$
using *star.circ-plus-one star-left-unfold-equal sup-assoc* **by** *presburger*
also have ... $= (w^+ \sqcap -F * e^T * e * -F \sqcap w^T * F * e^T * e * F * w) \sqcup (1 \sqcap -F * e^T * e * -F \sqcap w^T * F * e^T * e * F * w) \sqcup (w^{T+} \sqcap -F * e^T * e * -F \sqcap w^T * F * e^T * e * F * w)$
by (*simp add*: *inf-sup-distrib2*)
also have ... ≤ 1
using *11 12* **by** (*simp add*: *inf.coboundedI1*)
finally show $(w \sqcap ?E \sqcap ?F \sqcap ?G)^T * top * (w \sqcap ?E \sqcap ?F \sqcap ?G) \leq 1$
by *simp*
have $(w \sqcap -F) * (F \sqcap w^T) \leq w * w^T \sqcap -F * F$
by (*simp add*: *mult-isotone*)
also have ... $\leq 1 \sqcap -F$
using *assms*(*1*,*2*) *comp-inf.comp-isotone equivalence-comp-right-complement* **by** *auto*
also have ... $= bot$
using *assms*(*1*) *bot-unique pp-isotone pseudo-complement-pp* **by** *blast*
finally have *13*: $(w \sqcap -F) * (F \sqcap w^T) = bot$
by (*simp add*: *order.antisym*)
have $w \sqcap ?G \leq F * (w \sqcap ?G)$
by (*metis assms*(*1*) *mult-1-left mult-right-dist-sup sup.absorb-iff2*)
also have ... $\leq F * (w \sqcap ?G) * w^\star$
by (*metis eq-refl le-supE star.circ-back-loop-fixpoint*)
finally have *14*: $w \sqcap ?G \leq F * (w \sqcap ?G) * w^\star$
by *simp*
have $w \sqcap top * e * F \leq w * (e * F)^T * e * F$
by (*metis* (*no-types*) *comp-inf.star-slide dedekind-2 inf-left-commute inf-top-right mult-assoc*)
also have ... $\leq F$
using *3 assms*(*1*) **by** (*metis comp-associative conv-dist-comp mult-left-isotone preorder-idempotent*)
finally have $w \sqcap -F \leq -(top * e * F)$
using *order.trans p-shunting-swap pp-increasing* **by** *blast*
also have ... $= ?G$
by (*metis assms*(*3*) *comp-mapping-complement conv-dist-comp conv-involutive conv-top*)
finally have $(w \sqcap -F) * F * (w \sqcap ?G) = (w \sqcap -F \sqcap ?G) * F * (w \sqcap ?G)$
by (*simp add*: *inf.absorb1*)
also have ... $\leq (w \sqcap -F \sqcap ?G) * F * w$
by (*simp add*: *comp-isotone*)
also have ... $\leq (w \sqcap -F \sqcap ?G) *$ *forest-components* $(F \sqcap w) * w$
by (*simp add*: *assms*(*5*) *mult-left-isotone mult-right-isotone*)
also have ... $\leq (w \sqcap -F \sqcap ?G) * (F \sqcap w)^{T\star} * w^\star * w$
by (*simp add*: *mult-left-isotone mult-right-isotone star-isotone mult-assoc*)
also have ... $\leq (w \sqcap -F \sqcap ?G) * (F \sqcap w)^{T\star} * w^\star$
by (*simp add*: *comp-associative mult-right-isotone star.circ-plus-same*

star.left-plus-below-circ)
also have ... $= (w \sqcap -F \sqcap ?G) * w^\star \sqcup (w \sqcap -F \sqcap ?G) * (F \sqcap w)^{T+} * w^\star$
by (*metis comp-associative inf.sup-monoid.add-assoc mult-left-dist-sup star.circ-loop-fixpoint sup-commute*)
also have ... $\leq (w \sqcap -F \sqcap ?G) * w^\star \sqcup (w \sqcap -F \sqcap ?G) * (F \sqcap w)^T * top$
by (*metis mult-assoc top-greatest mult-right-isotone sup-right-isotone*)
also have ... $\leq (w \sqcap -F \sqcap ?G) * w^\star \sqcup (w \sqcap -F) * (F \sqcap w)^T * top$
using *inf.coubounded1 mult-left-isotone sup-right-isotone* **by** *blast*
also have ... $\leq (w \sqcap ?G) * w^\star \sqcup (w \sqcap -F) * (F \sqcap w)^T * top$
using *inf.sup-monoid.add-assoc inf.sup-right-isotone mult-left-isotone sup-commute sup-right-isotone* **by** *simp*
also have ... $= (w \sqcap ?G) * w^\star \sqcup (w \sqcap -F) * (F \sqcap w^T) * top$
by (*simp add*: *assms*(*1*) *conv-dist-inf*)
also have ... $\leq 1 * (w \sqcap ?G) * w^\star$
using *13* **by** *simp*
also have ... $\leq F * (w \sqcap ?G) * w^\star$
using *assms*(*1*) *mult-left-isotone* **by** *blast*
finally have *15*: $(w \sqcap -F) * F * (w \sqcap ?G) \leq F * (w \sqcap ?G) * w^\star$
by *simp*
have $(w \sqcap F) * F * (w \sqcap ?G) \leq F * F * (w \sqcap ?G)$
by (*simp add*: *mult-left-isotone*)
also have ... $= F * (w \sqcap ?G)$
by (*simp add*: *assms*(*1*) *preorder-idempotent*)
also have ... $\leq F * (w \sqcap ?G) * w^\star$
by (*metis eq-refl le-supE star.circ-back-loop-fixpoint*)
finally have $(w \sqcap F) * F * (w \sqcap ?G) \leq F * (w \sqcap ?G) * w^\star$
by *simp*
hence $((w \sqcap F) \sqcup (w \sqcap -F)) * F * (w \sqcap ?G) \leq F * (w \sqcap ?G) * w^\star$
using *15* **by** (*simp add*: *semiring.distrib-right*)
hence $w * F * (w \sqcap ?G) \leq F * (w \sqcap ?G) * w^\star$
by (*metis assms*(*4*) *maddux-3-11-pp*)
hence $w * F * (w \sqcap ?G) * w^\star \leq F * (w \sqcap ?G) * w^\star$
by (*metis* (*full-types*) *comp-associative mult-left-isotone star.circ-transitive-equal*)
hence $w^\star * (w \sqcap ?G) \leq F * (w \sqcap ?G) * w^\star$
using *14* **by** (*simp add*: *mult-assoc star-left-induct*)
hence *16*: $w^+ \sqcap ?G \leq F * (w \sqcap ?G) * w^\star$
by (*simp add*: *covector-comp-inf covector-mult-closed star.circ-plus-same*)
have *17*: $e^T * top * e^T \leq -F$
using *assms*(*8*) *le-bot triple-schroeder-p* **by** *simp*
hence $(top * e)^T * e^T \leq -F$
by (*simp add*: *conv-dist-comp*)
hence *18*: $e^T \leq ?G$
by (*metis assms*(*3*) *shunt-mapping conv-dist-comp conv-involutive conv-top*)
have $e^T \leq -F$
using *17* **by** (*simp add*: *assms*(*3*) *arc-top-arc*)
also have ... ≤ -1
by (*simp add*: *assms*(*1*) *p-antitone*)
finally have $e^T \leq w^\star \sqcap -1$

using *assms*(*9*) **by** *simp*
also have ... $\leq w^+$
using *shunting-var-p star-left-unfold-equal sup-commute* **by** *simp*
finally have $e^T \leq w^+ \sqcap ?G$
using *18* **by** *simp*
hence $e^T \leq F * (w \sqcap ?G) * w^\star$
using *16 order-trans* **by** *blast*
also have ... $= (F * w \sqcap ?G) * w^\star$
by (*simp add*: *comp-associative comp-inf-covector*)
finally have $e^T * top * e^T \leq (F * w \sqcap ?G) * w^\star$
by (*simp add*: *assms*(*3*) *arc-top-arc*)
hence $e^T * top * (e * top)^T \leq (F * w \sqcap ?G) * w^\star$
by (*metis conv-dist-comp conv-top vector-top-closed mult-assoc*)
hence $e^T * top \leq (F * w \sqcap ?G) * w^\star * e * top$
by (*metis assms*(*3*) *shunt-bijective mult-assoc*)
hence $(top * e)^T * top \leq (F * w \sqcap ?G) * w^\star * e * top$
by (*simp add*: *conv-dist-comp mult-assoc*)
hence $top \leq top * e * (F * w \sqcap ?G) * w^\star * e * top$
by (*metis assms*(*3*) *shunt-mapping conv-dist-comp conv-involutive conv-top mult-assoc*)
also have ... $= top * e * F * w * (w^\star * e * top \sqcap ?G^T)$
by (*metis comp-associative comp-inf-vector-1*)
also have ... $= top * (w \sqcap (top * e * F)^T) * (w^\star * e * top \sqcap ?G^T)$
by (*metis comp-inf-vector-1 inf-top.left-neutral*)
also have ... $= top * (w \sqcap ?F) * (w^\star * e * top \sqcap ?G^T)$
by (*simp add*: *assms*(*1*) *conv-dist-comp mult-assoc*)
also have ... $= top * (w \sqcap ?F) * (?E^T \sqcap ?G^T)$
by (*simp add*: *comp-associative conv-dist-comp conv-star-commute*)
also have ... $= top * (w \sqcap ?F \sqcap ?G) * ?E^T$
by (*simp add*: *comp-associative comp-inf-vector-1*)
also have ... $= top * (w \sqcap ?F \sqcap ?G \sqcap ?E) * top$
using *comp-inf-vector-1 mult-assoc* **by** *simp*
finally show $top * (w \sqcap ?E \sqcap ?F \sqcap ?G) * top = top$
by (*simp add*: *inf-commute inf-left-commute top-le*)
qed

lemma *kruskal-edge-arc-1*:
assumes $e \leq --h$
and $h \leq g$
and *symmetric* g
and *components* $g \leq$ *forest-components* w
and $w * e^T = bot$
shows $e^T \leq w^\star$
proof −
have $w^T * top \leq -(e^T * top)$
using *assms*(*5*) *schroeder-3-p vector-bot-closed mult-assoc* **by** *fastforce*
hence *1*: $w^T * top \sqcap e^T * top = bot$
using *pseudo-complement* **by** *simp*
have $e^T \leq e^T * top \sqcap --h^T$

using *assms*(*1*) *conv-complement conv-isotone top-right-mult-increasing* **by** *fastforce*
also have ... $\leq e^T * top \sqcap --g$
by (*metis assms*(*2*,*3*) *inf.sup-right-isotone pp-isotone conv-isotone*)
also have ... $\leq e^T * top \sqcap$ *components g*
using *inf.sup-right-isotone star.circ-increasing* **by** *simp*
also have ... $\leq e^T * top \sqcap$ *forest-components w*
using *assms*(*4*) *comp-inf.mult-right-isotone* **by** *simp*
also have ... $= (e^T * top \sqcap w^{T\star}) * w^\star$
by (*simp add*: *inf-assoc vector-export-comp*)
also have ... $= (e^T * top \sqcap 1) * w^\star \sqcup (e^T * top \sqcap w^{T+}) * w^\star$
by (*metis inf-sup-distrib1 semiring.distrib-right star-left-unfold-equal*)
also have ... $\leq w^\star \sqcup (e^T * top \sqcap w^{T+}) * w^\star$
by (*metis inf-le2 mult-1-left mult-left-isotone sup-left-isotone*)
also have ... $\leq w^\star \sqcup (e^T * top \sqcap w^T) * top$
using *comp-associative comp-inf.mult-right-isotone sup-right-isotone mult-right-isotone top.extremum vector-export-comp* **by** *presburger*
also have ... $= w^\star$
using *1 inf.sup-monoid.add-commute inf-vector-comp* **by** *simp*
finally show *?thesis*
by *simp*
qed

lemma *kruskal-edge-between-components-1*:
assumes *equivalence F*
and *mapping* (*top* $*$ *e*)
shows $F \leq -(w \sqcap top * e^T * w^{T\star} \sqcap F * e^T * top \sqcap top * e * -F)$
proof −
let $?d = w \sqcap top * e^T * w^{T\star} \sqcap F * e^T * top \sqcap top * e * -F$
have $?d \sqcap F \leq F * e^T * top \sqcap F$
by (*meson inf-le1 inf-le2 le-infI order-trans*)
also have ... $\leq (F * e^T * top)^T * F$
by (*simp add*: *mult-assoc vector-restrict-comp-conv*)
also have ... $= top * e * F * F$
by (*simp add*: *assms*(*1*) *comp-associative conv-dist-comp conv-star-commute*)
also have ... $= top * e * F$
using *assms*(*1*) *preorder-idempotent mult-assoc* **by** *fastforce*
finally have $?d \sqcap F \leq top * e * F \sqcap top * e * -F$
by (*simp add*: *le-infI1*)
also have ... $= top * e * F \sqcap -(top * e * F)$
using *assms*(*2*) *conv-dist-comp total-conv-surjective comp-mapping-complement* **by** *simp*
finally show *?thesis*
by (*metis inf-p le-bot p-antitone-iff pseudo-complement*)
qed

lemma *kruskal-edge-between-components-2*:
assumes *forest-components* $f \leq -d$
and *injective f*

and $f \sqcup f^T \leq w \sqcup w^T$
shows $f \sqcup f^T \leq (w \sqcap -d \sqcap -d^T) \sqcup (w^T \sqcap -d \sqcap -d^T)$
proof –
let *?F* = *forest-components f*
have $?F^T \leq -d^T$
using *assms*(*1*) *conv-complement conv-order* **by** *fastforce*
hence *1*: $?F \leq -d^T$
by (*simp add*: *conv-dist-comp conv-star-commute*)
have *equivalence ?F*
using *assms*(*2*) *forest-components-equivalence* **by** *simp*
hence $f \sqcup f^T \leq ?F$
by (*metis conv-dist-inf forest-components-increasing inf.absorb-iff2 sup.boundedI*)
also have ... $\leq -d \sqcap -d^T$
using *1 assms*(*1*) **by** *simp*
finally have $f \sqcup f^T \leq -d \sqcap -d^T$
by *simp*
thus *?thesis*
by (*metis assms*(*3*) *inf-sup-distrib2 le-inf-iff*)
qed

end

4.3 Related Structures

Stone algebras can be expanded to Stone-Kleene relation algebras by reusing some operations.

sublocale *stone-algebra* < *comp-inf*: *stone-kleene-relation-algebra* **where** *star* = λx . *top* **and** *one* = *top* **and** *times* = *inf* **and** *conv* = *id*
apply *unfold-locales*
by *simp*

Every bounded linear order can be expanded to a Stone algebra, which can be expanded to a Stone relation algebra, which can be expanded to a Stone-Kleene relation algebra.

class *linorder-stone-kleene-relation-algebra-expansion* = *linorder-stone-relation-algebra-expansion* + *star* +
assumes *star-def* [*simp*]: $x^\star = top$
begin

subclass *kleene-algebra*
apply *unfold-locales*
apply *simp*
apply (*simp add*: *min.coboundedI1 min.commute*)
by (*simp add*: *min.coboundedI1*)

subclass *stone-kleene-relation-algebra*
apply *unfold-locales*
by *simp*

end

A Kleene relation algebra is based on a relation algebra.

class *kleene-relation-algebra* = *relation-algebra* + *stone-kleene-relation-algebra*
begin

See https://arxiv.org/abs/2310.08946 for the following results *scc-1*-*scc-4*.

```
lemma scc-1:
  assumes 1 ⊓ y ≤ z
    and x^T * y ≤ y
    and y * z^T ≤ y
    and (x ⊓ y) * z ≤ z
   shows x⋆ ⊓ y ≤ z
proof −
  have x * (−y ⊔ z) ⊓ y = x * z ⊓ y
  proof (rule order.antisym)
    have x * (−y ⊔ z) ⊓ y ≤ x * ((−y ⊔ z) ⊓ x^T * y)
      by (simp add: dedekind-1)
    also have ... ≤ x * ((−y ⊔ z) ⊓ y)
      by (simp add: assms(2) le-infI2 mult-right-isotone)
    also have ... ≤ x * z
      by (simp add: inf.sup-monoid.add-commute mult-right-isotone)
    finally show x * (−y ⊔ z) ⊓ y ≤ x * z ⊓ y
      by simp
    show x * z ⊓ y ≤ x * (−y ⊔ z) ⊓ y
      by (simp add: inf-commute le-infI2 mult-right-isotone)
  qed
  also have ... ≤ (x ⊓ y * z^T) * z
    by (simp add: dedekind-2)
  also have ... ≤ (x ⊓ y) * z
    by (simp add: assms(3) le-infI2 mult-left-isotone)
  also have ... ≤ z
    by (simp add: assms(4))
  finally have 1: x * (−y ⊔ z) ⊓ y ≤ z
    .
  have (1 ⊔ x * (−y ⊔ z)) ⊓ y = (1 ⊓ y) ⊔ (x * (−y ⊔ z) ⊓ y)
    by (simp add: comp-inf.mult-right-dist-sup)
  also have ... ≤ z
    using 1 by (simp add: assms(1))
  finally have 1 ⊔ x * (−y ⊔ z) ≤ −y ⊔ z
    using shunt1 by blast
  hence x⋆ ≤ −y ⊔ z
    using star-left-induct by fastforce
  thus ?thesis
    by (simp add: shunt1)
qed
```

lemma *scc-2*:
  assumes $x^T * y \leq y$
    and $y * (x \sqcap y)^{\star T} \leq y$
  shows $x^\star \sqcap y \leq (x \sqcap y)^\star$
proof −
  have *1*: $1 \sqcap y \leq (x \sqcap y)^\star$
    by (*simp add*: *inf.coboundedI1 star.circ-reflexive*)
  have $(x \sqcap y) * (x \sqcap y)^\star \leq (x \sqcap y)^\star$
    by (*simp add*: *star.left-plus-below-circ*)
  thus *?thesis*
    using *1 assms scc-1* **by** *blast*
qed

lemma *scc-3*:
  $x^\star \sqcap x^{T\star} \leq (x \sqcap x^{T\star})^\star$
proof −
  have *1*: $x^T * x^{T\star} \leq x^{T\star}$
    by (*simp add*: *star.left-plus-below-circ*)
  have $x^{T\star} * (x \sqcap x^{T\star})^{\star T} \leq x^{T\star} * x^{\star T}$
    by (*simp add*: *star-isotone conv-isotone mult-right-isotone*)
  also have ... $= x^{T\star} * x^{T\star}$
    by (*simp add*: *conv-star-commute*)
  finally have $x^{T\star} * (x \sqcap x^{T\star})^{\star T} \leq x^{T\star}$
    by (*simp add*: *star.circ-transitive-equal*)
  thus *?thesis*
    using *1 scc-2* **by** *auto*
qed

lemma *scc-4*:
  $x^\star \sqcap x^{T\star} = (x \sqcap x^{T\star})^\star$
proof (*rule order.antisym*)
  show $x^\star \sqcap x^{T\star} \leq (x \sqcap x^{T\star})^\star$
    by (*simp add*: *scc-3*)
  have *1*: $(x \sqcap x^{T\star})^\star \leq x^\star$
    by (*simp add*: *star-isotone*)
  have $(x \sqcap x^{T\star})^\star \leq x^{T\star\star}$
    by (*simp add*: *star-isotone*)
  also have ... $= x^{T\star}$
    using *star-involutive* **by** *auto*
  finally show $(x \sqcap x^{T\star})^\star \leq x^\star \sqcap x^{T\star}$
    using *1* **by** *simp*
qed

end

class *stone-kleene-relation-algebra-tarski* = *stone-kleene-relation-algebra* + *stone-relation-algebra-tarski*

class *kleene-relation-algebra-tarski* = *kleene-relation-algebra* +

stone-kleene-relation-algebra-tarski
begin

subclass *relation-algebra-tarski* **..**

end

class *stone-kleene-relation-algebra-consistent* = *stone-kleene-relation-algebra* + *stone-relation-algebra-consistent*
begin

lemma *acyclic-reachable-different*:
  assumes *acyclic p bijective y* $x \leq p^+ * y$
  shows $x \neq y$
proof (*rule ccontr*)
  assume *1*: $\neg\ x \neq y$
  have $x * y^T \leq p^+$
    using *assms*(*2*,*3*) *shunt-bijective* **by** *blast*
  also have ... ≤ -1
    by (*simp add*: *assms*(*1*))
  finally show *False*
    using *1* **by** (*metis assms*(*2*) *dual-order.antisym le-supI2 mult-1-left order-char-1 point-not-bot schroeder-4-p semiring.mult-not-zero*)
qed

end

class *kleene-relation-algebra-consistent* = *kleene-relation-algebra* + *stone-kleene-relation-algebra-consistent*
begin

subclass *relation-algebra-consistent* **..**

end

class *stone-kleene-relation-algebra-tarski-consistent* = *stone-kleene-relation-algebra* + *stone-relation-algebra-tarski-consistent*
begin

subclass *stone-kleene-relation-algebra-tarski* **..**

subclass *stone-kleene-relation-algebra-consistent* **..**

end

class *kleene-relation-algebra-tarski-consistent* = *kleene-relation-algebra* + *stone-kleene-relation-algebra-tarski-consistent*
begin

subclass *relation-algebra-tarski-consistent* **..**

end

class *linorder-stone-kleene-relation-algebra-tarski-consistent-expansion* =
linorder-stone-kleene-relation-algebra-expansion + *non-trivial-bounded-order*
begin

subclass *stone-kleene-relation-algebra-tarski-consistent*
  apply *unfold-locales*
  by (*simp-all add*: *bot-not-top*)

end

end

5 Subalgebras of Kleene Relation Algebras

In this theory we show that the regular elements of a Stone-Kleene relation algebra form a Kleene relation subalgebra.

theory *Kleene-Relation-Subalgebras*

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

begin

instantiation *regular* :: (*stone-kleene-relation-algebra*) *kleene-relation-algebra*
begin

lift-definition *star-regular* :: *'a regular* ⇒ *'a regular* **is** *star*
  using *regular-closed-p regular-closed-star* **by** *blast*

instance
  apply *intro-classes*
  apply (*metis* (*mono-tags*, *lifting*) *star-regular.rep-eq less-eq-regular.rep-eq*
left-kleene-algebra-class.star-left-unfold one-regular.rep-eq simp-regular
sup-regular.rep-eq times-regular.rep-eq)
  apply (*metis* (*mono-tags*, *lifting*) *less-eq-regular.rep-eq*
left-kleene-algebra-class.star-left-induct simp-regular star-regular.rep-eq
sup-regular.rep-eq times-regular.rep-eq)
  apply (*metis* (*mono-tags*, *lifting*) *less-eq-regular.rep-eq*
strong-left-kleene-algebra-class.star-right-induct simp-regular star-regular.rep-eq
sup-regular.rep-eq times-regular.rep-eq)
  by *simp*

end

end

6 Matrix Kleene Algebras

This theory gives a matrix model of Stone-Kleene relation algebras. The main result is that matrices over Kleene algebras form Kleene algebras. The automata-based construction is due to Conway [7]. An implementation of the construction in Isabelle/HOL that extends [2] was given in [3] without a correctness proof.

For specifying the size of matrices, Isabelle/HOL's type system requires the use of types, not sets. This creates two issues when trying to implement Conway's recursive construction directly. First, the matrix size changes for recursive calls, which requires dependent types. Second, some submatrices used in the construction are not square, which requires typed Kleene algebras [14], that is, categories of Kleene algebras.

Because these instruments are not available in Isabelle/HOL, we use square matrices with a constant size given by the argument of the Kleene star operation. Smaller, possibly rectangular submatrices are identified by two lists of indices: one for the rows to include and one for the columns to include. Lists are used to make recursive calls deterministic; otherwise sets would be sufficient.

theory *Matrix-Kleene-Algebras*

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

begin

6.1 Matrix Restrictions

In this section we develop a calculus of matrix restrictions. The restriction of a matrix to specific row and column indices is implemented by the following function, which keeps the size of the matrix and sets all unused entries to *bot*.

definition *restrict-matrix* :: *'a list* ⇒ (*'a*,*'b*::*bot*) *square* ⇒ *'a list* ⇒ (*'a*,*'b*) *square* (‹- ⟨-⟩ -› [*90*,*41*,*90*] *91*)
  where *restrict-matrix as f bs* = (λ(*i*,*j*) . *if i* ∈ *set as* ∧ *j* ∈ *set bs then f* (*i*,*j*) *else bot*)

The following function captures Conway's automata-based construction of the Kleene star of a matrix. An index k is chosen and s contains all other indices. The matrix is split into four submatrices a, b, c, d including/not including row/column k. Four matrices are computed containing the entries given by Conway's construction. These four matrices are added to obtain the result. All matrices involved in the function have the same size, but matrix restriction is used to set irrelevant entries to *bot*.

primrec *star-matrix'* :: *'a list* ⇒ (*'a*,*'b*::{*star*,*times*,*bounded-semilattice-sup-bot*}) *square* ⇒ (*'a*,*'b*) *square* **where**

$star\text{-}matrix'\ Nil\ g = mbot\ |$
$star\text{-}matrix'\ (k\#s)\ g = ($
  $let\ r = [k]\ in$
  $let\ a = r\langle g\rangle r\ in$
  $let\ b = r\langle g\rangle s\ in$
  $let\ c = s\langle g\rangle r\ in$
  $let\ d = s\langle g\rangle s\ in$
  $let\ as = r\langle star\ o\ a\rangle r\ in$
  $let\ ds = star\text{-}matrix'\ s\ d\ in$
  $let\ e = a \oplus b \odot ds \odot c\ in$
  $let\ es = r\langle star\ o\ e\rangle r\ in$
  $let\ f = d \oplus c \odot as \odot b\ in$
  $let\ fs = star\text{-}matrix'\ s\ f\ in$
  $es \oplus as \odot b \odot fs \oplus ds \odot c \odot es \oplus fs$
$)$

The Kleene star of the whole matrix is obtained by taking as indices all elements of the underlying type $'a$. This is conveniently supplied by the *enum* class.

fun $star\text{-}matrix :: ('a::enum,'b::\{star,times,bounded\text{-}semilattice\text{-}sup\text{-}bot\})\ square \Rightarrow ('a,'b)\ square\ (‹\text{-}^{\odot}›\ [100]\ 100)$ **where** $star\text{-}matrix\ f = star\text{-}matrix'\ (enum\text{-}class.enum::'a\ list)\ f$

The following lemmas deconstruct matrices with non-empty restrictions.

lemma *restrict-empty-left*:
  $[]\langle f\rangle ls = mbot$
  by (*unfold restrict-matrix-def bot-matrix-def*) *auto*

lemma *restrict-empty-right*:
  $ks\langle f\rangle[] = mbot$
  by (*unfold restrict-matrix-def bot-matrix-def*) *auto*

lemma *restrict-nonempty-left*:
  fixes $f :: ('a,'b::bounded\text{-}semilattice\text{-}sup\text{-}bot)\ square$
  shows $(k\#ks)\langle f\rangle ls = [k]\langle f\rangle ls \oplus ks\langle f\rangle ls$
  by (*unfold restrict-matrix-def sup-matrix-def*) *auto*

lemma *restrict-nonempty-right*:
  fixes $f :: ('a,'b::bounded\text{-}semilattice\text{-}sup\text{-}bot)\ square$
  shows $ks\langle f\rangle(l\#ls) = ks\langle f\rangle[l] \oplus ks\langle f\rangle ls$
  by (*unfold restrict-matrix-def sup-matrix-def*) *auto*

lemma *restrict-nonempty*:
  fixes $f :: ('a,'b::bounded\text{-}semilattice\text{-}sup\text{-}bot)\ square$
  shows $(k\#ks)\langle f\rangle(l\#ls) = [k]\langle f\rangle[l] \oplus [k]\langle f\rangle ls \oplus ks\langle f\rangle[l] \oplus ks\langle f\rangle ls$
  by (*unfold restrict-matrix-def sup-matrix-def*) *auto*

The following predicate captures that two index sets are disjoint. This has consequences for composition and the unit matrix.

abbreviation *disjoint ks ls* ≡ ¬ (∃ *x* . *x* ∈ *set ks* ∧ *x* ∈ *set ls*)

lemma *times-disjoint*:
  fixes *f g* :: (*'a*,*'b*::*idempotent-semiring*) *square*
  assumes *disjoint ls ms*
    shows *ks*⟨*f*⟩*ls* ⊙ *ms*⟨*g*⟩*ns* = *mbot*
proof (*rule ext*, *rule prod-cases*)
  fix *i j*
  have (*ks*⟨*f*⟩*ls* ⊙ *ms*⟨*g*⟩*ns*) (*i*,*j*) = (⨆$_k$ (*ks*⟨*f*⟩*ls*) (*i*,*k*) ∗ (*ms*⟨*g*⟩*ns*) (*k*,*j*))
    by (*simp add*: *times-matrix-def*)
  also have ... = (⨆$_k$ (*if i* ∈ *set ks* ∧ *k* ∈ *set ls then f* (*i*,*k*) *else bot*)
    ∗ (*if k* ∈ *set ms* ∧ *j* ∈ *set ns then g* (*k*,*j*) *else bot*))
    by (*simp add*: *restrict-matrix-def*)
  also have ... = (⨆$_k$ *if k* ∈ *set ms* ∧ *j* ∈ *set ns then bot* ∗ *g* (*k*,*j*)
    else (*if i* ∈ *set ks* ∧ *k* ∈ *set ls then f* (*i*,*k*) *else bot*) ∗ *bot*)
    using *assms* **by** (*auto intro*: *sup-monoid.sum.cong*)
  also have ... = (⨆$_{(k::'a)}$ *bot*)
    by (*simp add*: *sup-monoid.sum.neutral*)
  also have ... = *bot*
    by (*simp add*: *eq-iff le-funI*)
  also have ... = *mbot* (*i*,*j*)
    by (*simp add*: *bot-matrix-def*)
  finally show (*ks*⟨*f*⟩*ls* ⊙ *ms*⟨*g*⟩*ns*) (*i*,*j*) = *mbot* (*i*,*j*)
    .
qed

lemma *one-disjoint*:
  assumes *disjoint ks ls*
    shows *ks*⟨(*mone*::(*'a*,*'b*::*idempotent-semiring*) *square*)⟩*ls* = *mbot*
proof (*rule ext*, *rule prod-cases*)
  let *?o* = *mone*::(*'a*,*'b*) *square*
  fix *i j*
  have (*ks*⟨*?o*⟩*ls*) (*i*,*j*) = (*if i* ∈ *set ks* ∧ *j* ∈ *set ls then if i* = *j then 1 else bot else bot*)
    by (*simp add*: *restrict-matrix-def one-matrix-def*)
  also have ... = *bot*
    using *assms* **by** *auto*
  also have ... = *mbot* (*i*,*j*)
    by (*simp add*: *bot-matrix-def*)
  finally show (*ks*⟨*?o*⟩*ls*) (*i*,*j*) = *mbot* (*i*,*j*)
    .
qed

The following predicate captures that an index set is a subset of another index set. This has consequences for repeated restrictions.

abbreviation *is-sublist ks ls* ≡ *set ks* ⊆ *set ls*

lemma *restrict-sublist*:
  assumes *is-sublist ls ks*

and *is-sublist ms ns*
shows $ls\langle ks\langle f\rangle ns\rangle ms = ls\langle f\rangle ms$
proof (*rule ext*, *rule prod-cases*)
fix *i j*
show $(ls\langle ks\langle f\rangle ns\rangle ms)\ (i,j) = (ls\langle f\rangle ms)\ (i,j)$
proof (*cases* $i \in set\ ls \wedge j \in set\ ms$)
case *True*
with *assms* **show** *?thesis*
by (*auto simp add*: *restrict-matrix-def*)
next
case *False*
with *assms* **show** *?thesis*
by (*auto simp add*: *restrict-matrix-def*)
qed
qed

lemma *restrict-superlist*:
assumes *is-sublist ls ks*
and *is-sublist ms ns*
shows $ks\langle ls\langle f\rangle ms\rangle ns = ls\langle f\rangle ms$
proof (*rule ext*, *rule prod-cases*)
fix *i j*
show $(ks\langle ls\langle f\rangle ms\rangle ns)\ (i,j) = (ls\langle f\rangle ms)\ (i,j)$
proof (*cases* $i \in set\ ls \wedge j \in set\ ms$)
case *True*
with *assms* **show** *?thesis*
by (*auto simp add*: *restrict-matrix-def*)
next
case *False*
with *assms* **show** *?thesis*
by (*auto simp add*: *restrict-matrix-def*)
qed
qed

The following lemmas give the sizes of the results of some matrix operations.

lemma *restrict-sup*:
fixes *f g* :: (*'a*,*'b*::*bounded-semilattice-sup-bot*) *square*
shows $ks\langle f \oplus g\rangle ls = ks\langle f\rangle ls \oplus ks\langle g\rangle ls$
by (*unfold restrict-matrix-def sup-matrix-def*) *auto*

lemma *restrict-times*:
fixes *f g* :: (*'a*,*'b*::*idempotent-semiring*) *square*
shows $ks\langle ks\langle f\rangle ls \odot ls\langle g\rangle ms\rangle ms = ks\langle f\rangle ls \odot ls\langle g\rangle ms$
proof (*rule ext*, *rule prod-cases*)
fix *i j*
have $(ks\langle (ks\langle f\rangle ls \odot ls\langle g\rangle ms)\rangle ms)\ (i,j) = (if\ i \in set\ ks \wedge j \in set\ ms\ then\ (\bigsqcup_k (ks\langle f\rangle ls)\ (i,k) * (ls\langle g\rangle ms)\ (k,j))\ else\ bot)$
by (*simp add*: *times-matrix-def restrict-matrix-def*)

```
  also have ... = (if i ∈ set ks ∧ j ∈ set ms then (⨆k (if i ∈ set ks ∧ k ∈ set ls
then f (i,k) else bot) ∗ (if k ∈ set ls ∧ j ∈ set ms then g (k,j) else bot)) else bot)
    by (simp add: restrict-matrix-def)
  also have ... = (if i ∈ set ks ∧ j ∈ set ms then (⨆k if k ∈ set ls then f (i,k) ∗
g (k,j) else bot) else bot)
    by (auto intro: sup-monoid.sum.cong)
  also have ... = (⨆k if i ∈ set ks ∧ j ∈ set ms then (if k ∈ set ls then f (i,k) ∗
g (k,j) else bot) else bot)
    by auto
  also have ... = (⨆k (if i ∈ set ks ∧ k ∈ set ls then f (i,k) else bot) ∗ (if k ∈ set
ls ∧ j ∈ set ms then g (k,j) else bot))
    by (auto intro: sup-monoid.sum.cong)
  also have ... = (⨆k (ks⟨f⟩ls) (i,k) ∗ (ls⟨g⟩ms) (k,j))
    by (simp add: restrict-matrix-def)
  also have ... = (ks⟨f⟩ls ⊙ ls⟨g⟩ms) (i,j)
    by (simp add: times-matrix-def)
  finally show (ks⟨(ks⟨f⟩ls ⊙ ls⟨g⟩ms)⟩ms) (i,j) = (ks⟨f⟩ls ⊙ ls⟨g⟩ms) (i,j)
    .
qed

lemma restrict-star:
  fixes g :: ('a,'b::kleene-algebra) square
  shows t⟨star-matrix' t g⟩t = star-matrix' t g
proof (induct arbitrary: g rule: list.induct)
  case Nil show ?case
    by (simp add: restrict-empty-left)
next
  case (Cons k s)
  let ?t = k#s
  assume ⋀g::('a,'b) square . s⟨star-matrix' s g⟩s = star-matrix' s g
  then have 1: ?t⟨star-matrix' s g⟩?t = star-matrix' s g for g :: ‹('a, 'b) square›
    using restrict-superlist [of s ‹k # s› s ‹k # s› ‹star-matrix' s g›]
    by auto
  show ?t⟨star-matrix' ?t g⟩?t = star-matrix' ?t g
  proof −
    let ?r = [k]
    let ?a = ?r⟨g⟩?r
    let ?b = ?r⟨g⟩s
    let ?c = s⟨g⟩?r
    let ?d = s⟨g⟩s
    let ?as = ?r⟨star o ?a⟩?r
    let ?ds = star-matrix' s ?d
    let ?e = ?a ⊕ ?b ⊙ ?ds ⊙ ?c
    let ?es = ?r⟨star o ?e⟩?r
    let ?f = ?d ⊕ ?c ⊙ ?as ⊙ ?b
    let ?fs = star-matrix' s ?f
    have 2: ?t⟨?as⟩?t = ?as ∧ ?t⟨?b⟩?t = ?b ∧ ?t⟨?c⟩?t = ?c ∧ ?t⟨?es⟩?t = ?es
      by (simp add: restrict-superlist subset-eq)
    have 3: ?t⟨?ds⟩?t = ?ds ∧ ?t⟨?fs⟩?t = ?fs
```

using *1* **by** *simp*
have *4*: $?t\langle ?t\langle ?as\rangle ?t \odot ?t\langle ?b\rangle ?t \odot ?t\langle ?fs\rangle ?t\rangle ?t = ?t\langle ?as\rangle ?t \odot ?t\langle ?b\rangle ?t \odot ?t\langle ?fs\rangle ?t$
by (*metis* (*no-types*) *restrict-times*)
have *5*: $?t\langle ?t\langle ?ds\rangle ?t \odot ?t\langle ?c\rangle ?t \odot ?t\langle ?es\rangle ?t\rangle ?t = ?t\langle ?ds\rangle ?t \odot ?t\langle ?c\rangle ?t \odot ?t\langle ?es\rangle ?t$
by (*metis* (*no-types*) *restrict-times*)
have $?t\langle star\text{-}matrix'\ ?t\ g\rangle ?t = ?t\langle ?es \oplus ?as \odot ?b \odot ?fs \oplus ?ds \odot ?c \odot ?es \oplus ?fs\rangle ?t$
by (*metis star-matrix'.simps*(*2*))
also have ... $= ?t\langle ?es\rangle ?t \oplus ?t\langle ?as \odot ?b \odot ?fs\rangle ?t \oplus ?t\langle ?ds \odot ?c \odot ?es\rangle ?t \oplus ?t\langle ?fs\rangle ?t$
by (*simp add*: *restrict-sup*)
also have ... $= ?es \oplus ?as \odot ?b \odot ?fs \oplus ?ds \odot ?c \odot ?es \oplus ?fs$
using *2 3 4 5* **by** *simp*
also have ... = *star-matrix' ?t g*
by (*metis star-matrix'.simps*(*2*))
finally show *?thesis*
.
qed
qed

lemma *restrict-one*:
assumes $\neg\ k \in set\ ks$
shows $(k\#ks)\langle(mone::('a,'b::idempotent\text{-}semiring)\ square)\rangle(k\#ks) = [k]\langle mone\rangle[k] \oplus ks\langle mone\rangle ks$
by (*subst restrict-nonempty*) (*simp add*: *assms one-disjoint*)

lemma *restrict-one-left-unit*:
$ks\langle(mone::('a::finite,'b::idempotent\text{-}semiring)\ square)\rangle ks \odot ks\langle f\rangle ls = ks\langle f\rangle ls$
proof (*rule ext*, *rule prod-cases*)
let $?o = mone::('a,'b::idempotent\text{-}semiring)\ square$
fix *i j*
have $(ks\langle ?o\rangle ks \odot ks\langle f\rangle ls)\ (i,j) = (\bigsqcup_k\ (ks\langle ?o\rangle ks)\ (i,k) * (ks\langle f\rangle ls)\ (k,j))$
by (*simp add*: *times-matrix-def*)
also have ... $= (\bigsqcup_k\ (if\ i \in set\ ks \wedge k \in set\ ks\ then\ ?o\ (i,k)\ else\ bot) * (if\ k \in set\ ks \wedge j \in set\ ls\ then\ f\ (k,j)\ else\ bot))$
by (*simp add*: *restrict-matrix-def*)
also have ... $= (\bigsqcup_k\ (if\ i \in set\ ks \wedge k \in set\ ks\ then\ (if\ i = k\ then\ 1\ else\ bot)\ else\ bot) * (if\ k \in set\ ks \wedge j \in set\ ls\ then\ f\ (k,j)\ else\ bot))$
by (*unfold one-matrix-def*) *auto*
also have ... $= (\bigsqcup_k\ (if\ i = k\ then\ (if\ i \in set\ ks\ then\ 1\ else\ bot)\ else\ bot) * (if\ k \in set\ ks \wedge j \in set\ ls\ then\ f\ (k,j)\ else\ bot))$
by (*auto intro*: *sup-monoid.sum.cong*)
also have ... $= (\bigsqcup_k\ if\ i = k\ then\ (if\ i \in set\ ks\ then\ 1\ else\ bot) * (if\ i \in set\ ks \wedge j \in set\ ls\ then\ f\ (i,j)\ else\ bot)\ else\ bot)$
by (*rule sup-monoid.sum.cong*) *simp-all*
also have ... $= (if\ i \in set\ ks\ then\ 1\ else\ bot) * (if\ i \in set\ ks \wedge j \in set\ ls\ then\ f\ (i,j)\ else\ bot)$

by *simp*
also have ... = (*if* $i \in$ *set ks* $\wedge$ $j \in$ *set ls then f* (i,j) *else bot*)
by *simp*
also have ... = $(ks\langle f\rangle ls)$ (i,j)
by (*simp add*: *restrict-matrix-def*)
finally show $(ks\langle ?o\rangle ks \odot ks\langle f\rangle ls)$ (i,j) = $(ks\langle f\rangle ls)$ (i,j)
.
qed

The following lemmas consider restrictions to singleton index sets.

lemma *restrict-singleton*:
$([k]\langle f\rangle[l])$ (i,j) = (*if* $i = k \wedge j = l$ *then f* (i,j) *else bot*)
by (*simp add*: *restrict-matrix-def*)

lemma *restrict-singleton-list*:
$([k]\langle f\rangle ls)$ (i,j) = (*if* $i = k \wedge j \in$ *set ls then f* (i,j) *else bot*)
by (*simp add*: *restrict-matrix-def*)

lemma *restrict-list-singleton*:
$(ks\langle f\rangle[l])$ (i,j) = (*if* $i \in$ *set ks* $\wedge$ $j = l$ *then f* (i,j) *else bot*)
by (*simp add*: *restrict-matrix-def*)

lemma *restrict-singleton-product*:
fixes *f g* :: ('*a*::*finite*,'*b*::*kleene-algebra*) *square*
shows $([k]\langle f\rangle[l] \odot [m]\langle g\rangle[n])$ (i,j) = (*if* $i = k \wedge l = m \wedge j = n$ *then f* $(i,l) * g$ (m,j) *else bot*)
proof −
have $([k]\langle f\rangle[l] \odot [m]\langle g\rangle[n])$ (i,j) = $(\bigsqcup_h ([k]\langle f\rangle[l])\ (i,h) * ([m]\langle g\rangle[n])\ (h,j))$
by (*simp add*: *times-matrix-def*)
also have ... = $(\bigsqcup_h$ (*if* $i = k \wedge h = l$ *then f* (i,h) *else bot*) $*$ (*if* $h = m \wedge j = n$ *then g* (h,j) *else bot*))
by (*simp add*: *restrict-singleton*)
also have ... = $(\bigsqcup_h$ *if* $h = l$ *then* (*if* $i = k$ *then f* (i,h) *else bot*) $*$ (*if* $h = m \wedge j = n$ *then g* (h,j) *else bot*) *else bot*)
by (*rule sup-monoid.sum.cong*) *auto*
also have ... = (*if* $i = k$ *then f* (i,l) *else bot*) $*$ (*if* $l = m \wedge j = n$ *then g* (l,j) *else bot*)
by *simp*
also have ... = (*if* $i = k \wedge l = m \wedge j = n$ *then f* $(i,l) * g$ (m,j) *else bot*)
by *simp*
finally show *?thesis*
.
qed

The Kleene star unfold law holds for matrices with a single entry on the diagonal.

lemma *restrict-star-unfold*:
$[l]\langle(mone::('a::finite,'b::kleene\text{-}algebra)\ square)\rangle[l] \oplus [l]\langle f\rangle[l] \odot [l]\langle star\ o\ f\rangle[l]$ = $[l]\langle star\ o\ f\rangle[l]$

proof (*rule ext*, *rule prod-cases*)
  let *?o* = *mone*::(*'a*,*'b*::*kleene-algebra*) *square*
  fix *i j*
  have $([l]\langle ?o\rangle[l] \oplus [l]\langle f\rangle[l] \odot [l]\langle star\ o\ f\rangle[l])\ (i,j) = ([l]\langle ?o\rangle[l])\ (i,j) \sqcup ([l]\langle f\rangle[l] \odot [l]\langle star\ o\ f\rangle[l])\ (i,j)$
    by (*simp add*: *sup-matrix-def*)
  also have ... $= ([l]\langle ?o\rangle[l])\ (i,j) \sqcup (\bigsqcup_k\ ([l]\langle f\rangle[l])\ (i,k) * ([l]\langle star\ o\ f\rangle[l])\ (k,j))$
    by (*simp add*: *times-matrix-def*)
  also have ... $= ([l]\langle ?o\rangle[l])\ (i,j) \sqcup (\bigsqcup_k\ (if\ i = l \wedge k = l\ then\ f\ (i,k)\ else\ bot) * (if\ k = l \wedge j = l\ then\ (f\ (k,j))^\star\ else\ bot))$
    by (*simp add*: *restrict-singleton o-def*)
  also have ... $= ([l]\langle ?o\rangle[l])\ (i,j) \sqcup (\bigsqcup_k\ if\ k = l\ then\ (if\ i = l\ then\ f\ (i,k)\ else\ bot) * (if\ j = l\ then\ (f\ (k,j))^\star\ else\ bot)\ else\ bot)$
    apply (*rule arg-cong2*[**where** *f*=*sup*])
    apply *simp*
    by (*rule sup-monoid.sum.cong*) *auto*
  also have ... $= ([l]\langle ?o\rangle[l])\ (i,j) \sqcup (if\ i = l\ then\ f\ (i,l)\ else\ bot) * (if\ j = l\ then\ (f\ (l,j))^\star\ else\ bot)$
    by *simp*
  also have ... $= (if\ i = l \wedge j = l\ then\ 1 \sqcup f\ (l,l) * (f\ (l,l))^\star\ else\ bot)$
    by (*simp add*: *restrict-singleton one-matrix-def*)
  also have ... $= (if\ i = l \wedge j = l\ then\ (f\ (l,l))^\star\ else\ bot)$
    by (*simp add*: *star-left-unfold-equal*)
  also have ... $= ([l]\langle star\ o\ f\rangle[l])\ (i,j)$
    by (*simp add*: *restrict-singleton o-def*)
  finally show $([l]\langle ?o\rangle[l] \oplus [l]\langle f\rangle[l] \odot [l]\langle star\ o\ f\rangle[l])\ (i,j) = ([l]\langle star\ o\ f\rangle[l])\ (i,j)$
    .
qed

lemma *restrict-all*:
  $enum\text{-}class.enum\langle f\rangle enum\text{-}class.enum = f$
  by (*simp add*: *restrict-matrix-def enum-UNIV*)

The following shows the various components of a matrix product. It is essentially a recursive implementation of the product.

lemma *restrict-nonempty-product*:
  fixes *f g* :: (*'a*::*finite*,*'b*::*idempotent-semiring*) *square*
  assumes $\neg\ l \in set\ ls$
    shows $(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms) = ([k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle[m]) \oplus ([k]\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms) \oplus (ks\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle[m]) \oplus (ks\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms)$
proof −
  have $(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms) = ([k]\langle f\rangle[l] \oplus [k]\langle f\rangle ls \oplus ks\langle f\rangle[l] \oplus ks\langle f\rangle ls) \odot ([l]\langle g\rangle[m] \oplus [l]\langle g\rangle ms \oplus ls\langle g\rangle[m] \oplus ls\langle g\rangle ms)$
    by (*metis restrict-nonempty*)
  also have ... $= [k]\langle f\rangle[l] \odot ([l]\langle g\rangle[m] \oplus [l]\langle g\rangle ms \oplus ls\langle g\rangle[m] \oplus ls\langle g\rangle ms) \oplus [k]\langle f\rangle ls \odot ([l]\langle g\rangle[m] \oplus [l]\langle g\rangle ms \oplus ls\langle g\rangle[m] \oplus ls\langle g\rangle ms) \oplus ks\langle f\rangle[l] \odot ([l]\langle g\rangle[m] \oplus [l]\langle g\rangle ms \oplus ls\langle g\rangle[m] \oplus ls\langle g\rangle ms) \oplus ks\langle f\rangle ls \odot ([l]\langle g\rangle[m] \oplus [l]\langle g\rangle ms \oplus ls\langle g\rangle[m] \oplus ls\langle g\rangle ms)$
    by (*simp add*: *matrix-idempotent-semiring.mult-right-dist-sup*)

also have ... $= ([k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle[l] \odot ls\langle g\rangle[m] \oplus [k]\langle f\rangle[l] \odot ls\langle g\rangle ms) \oplus ([k]\langle f\rangle ls \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms) \oplus (ks\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle[l] \odot ls\langle g\rangle[m] \oplus ks\langle f\rangle[l] \odot ls\langle g\rangle ms) \oplus (ks\langle f\rangle ls \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle ls \odot ls\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms)$
by (*simp add*: *matrix-idempotent-semiring.mult-left-dist-sup*)
also have ... $= ([k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle[l] \odot [l]\langle g\rangle ms) \oplus ([k]\langle f\rangle ls \odot ls\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms) \oplus (ks\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle[l] \odot [l]\langle g\rangle ms) \oplus (ks\langle f\rangle ls \odot ls\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms)$
using *assms* **by** (*simp add*: *times-disjoint*)
also have ... $= ([k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle[m]) \oplus ([k]\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms) \oplus (ks\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle[m]) \oplus (ks\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms)$
by (*simp add*: *matrix-bounded-semilattice-sup-bot.sup-monoid.add-assoc matrix-semilattice-sup.sup-left-commute*)
finally show *?thesis*
.
qed

Equality of matrices is componentwise.

lemma *restrict-nonempty-eq*:
$(k\#ks)\langle f\rangle(l\#ls) = (k\#ks)\langle g\rangle(l\#ls) \longleftrightarrow [k]\langle f\rangle[l] = [k]\langle g\rangle[l] \wedge [k]\langle f\rangle ls = [k]\langle g\rangle ls \wedge ks\langle f\rangle[l] = ks\langle g\rangle[l] \wedge ks\langle f\rangle ls = ks\langle g\rangle ls$
proof
assume *1*: $(k\#ks)\langle f\rangle(l\#ls) = (k\#ks)\langle g\rangle(l\#ls)$
have *2*: *is-sublist* $[k]$ $(k\#ks)$ $\wedge$ *is-sublist* ks $(k\#ks)$ $\wedge$ *is-sublist* $[l]$ $(l\#ls)$ $\wedge$ *is-sublist* ls $(l\#ls)$
by *auto*
hence $[k]\langle f\rangle[l] = [k]\langle(k\#ks)\langle f\rangle(l\#ls)\rangle[l] \wedge [k]\langle f\rangle ls = [k]\langle(k\#ks)\langle f\rangle(l\#ls)\rangle ls \wedge ks\langle f\rangle[l] = ks\langle(k\#ks)\langle f\rangle(l\#ls)\rangle[l] \wedge ks\langle f\rangle ls = ks\langle(k\#ks)\langle f\rangle(l\#ls)\rangle ls$
by (*simp add*: *restrict-sublist*)
thus $[k]\langle f\rangle[l] = [k]\langle g\rangle[l] \wedge [k]\langle f\rangle ls = [k]\langle g\rangle ls \wedge ks\langle f\rangle[l] = ks\langle g\rangle[l] \wedge ks\langle f\rangle ls = ks\langle g\rangle ls$
using *1 2* **by** (*simp add*: *restrict-sublist*)
next
assume *3*: $[k]\langle f\rangle[l] = [k]\langle g\rangle[l] \wedge [k]\langle f\rangle ls = [k]\langle g\rangle ls \wedge ks\langle f\rangle[l] = ks\langle g\rangle[l] \wedge ks\langle f\rangle ls = ks\langle g\rangle ls$
show $(k\#ks)\langle f\rangle(l\#ls) = (k\#ks)\langle g\rangle(l\#ls)$
proof (*rule ext*, *rule prod-cases*)
fix *i j*
have *4*: $f\ (k,l) = g\ (k,l)$
using *3* **by** (*metis restrict-singleton*)
have *5*: $j \in set\ ls \Longrightarrow f\ (k,j) = g\ (k,j)$
using *3* **by** (*metis restrict-singleton-list*)
have *6*: $i \in set\ ks \Longrightarrow f\ (i,l) = g\ (i,l)$
using *3* **by** (*metis restrict-list-singleton*)
have $(ks\langle f\rangle ls)\ (i,j) = (ks\langle g\rangle ls)\ (i,j)$
using *3* **by** *simp*
hence *7*: $i \in set\ ks \Longrightarrow j \in set\ ls \Longrightarrow f\ (i,j) = g\ (i,j)$

by (*simp add*: *restrict-matrix-def*)
have $((k\#ks)\langle f\rangle(l\#ls))\ (i,j) = (\mathit{if}\ (i = k \lor i \in \mathit{set}\ ks) \land (j = l \lor j \in \mathit{set}\ ls)\ \mathit{then}\ f\ (i,j)\ \mathit{else}\ \mathit{bot})$
by (*simp add*: *restrict-matrix-def*)
also have ... $= (\mathit{if}\ i = k \land j = l\ \mathit{then}\ f\ (i,j)\ \mathit{else}\ \mathit{if}\ i = k \land j \in \mathit{set}\ ls\ \mathit{then}\ f\ (i,j)\ \mathit{else}\ \mathit{if}\ i \in \mathit{set}\ ks \land j = l\ \mathit{then}\ f\ (i,j)\ \mathit{else}\ \mathit{if}\ i \in \mathit{set}\ ks \land j \in \mathit{set}\ ls\ \mathit{then}\ f\ (i,j)\ \mathit{else}\ \mathit{bot})$
by *auto*
also have ... $= (\mathit{if}\ i = k \land j = l\ \mathit{then}\ g\ (i,j)\ \mathit{else}\ \mathit{if}\ i = k \land j \in \mathit{set}\ ls\ \mathit{then}\ g\ (i,j)\ \mathit{else}\ \mathit{if}\ i \in \mathit{set}\ ks \land j = l\ \mathit{then}\ g\ (i,j)\ \mathit{else}\ \mathit{if}\ i \in \mathit{set}\ ks \land j \in \mathit{set}\ ls\ \mathit{then}\ g\ (i,j)\ \mathit{else}\ \mathit{bot})$
using *4 5 6 7* **by** *simp*
also have ... $= (\mathit{if}\ (i = k \lor i \in \mathit{set}\ ks) \land (j = l \lor j \in \mathit{set}\ ls)\ \mathit{then}\ g\ (i,j)\ \mathit{else}\ \mathit{bot})$
by *auto*
also have ... $= ((k\#ks)\langle g\rangle(l\#ls))\ (i,j)$
by (*simp add*: *restrict-matrix-def*)
finally show $((k\#ks)\langle f\rangle(l\#ls))\ (i,j) = ((k\#ks)\langle g\rangle(l\#ls))\ (i,j)$
.
qed
qed

Inequality of matrices is componentwise.

lemma *restrict-nonempty-less-eq*:
fixes $f\ g :: ('a,'b{::}\mathit{idempotent\text{-}semiring})\ \mathit{square}$
shows $(k\#ks)\langle f\rangle(l\#ls) \preceq (k\#ks)\langle g\rangle(l\#ls) \longleftrightarrow [k]\langle f\rangle[l] \preceq [k]\langle g\rangle[l] \land [k]\langle f\rangle ls \preceq [k]\langle g\rangle ls \land ks\langle f\rangle[l] \preceq ks\langle g\rangle[l] \land ks\langle f\rangle ls \preceq ks\langle g\rangle ls$
by (*unfold matrix-semilattice-sup.sup.order-iff*) (*metis* (*no-types*, *lifting*) *restrict-nonempty-eq restrict-sup*)

The following lemmas treat repeated restrictions to disjoint index sets.

lemma *restrict-disjoint-left*:
assumes *disjoint ks ms*
shows $ms\langle ks\langle f\rangle ls\rangle ns = \mathit{mbot}$
proof (*rule ext*, *rule prod-cases*)
fix *i j*
have $(ms\langle ks\langle f\rangle ls\rangle ns)\ (i,j) = (\mathit{if}\ i \in \mathit{set}\ ms \land j \in \mathit{set}\ ns\ \mathit{then}\ \mathit{if}\ i \in \mathit{set}\ ks \land j \in \mathit{set}\ ls\ \mathit{then}\ f\ (i,j)\ \mathit{else}\ \mathit{bot}\ \mathit{else}\ \mathit{bot})$
by (*auto simp add*: *restrict-matrix-def*)
thus $(ms\langle ks\langle f\rangle ls\rangle ns)\ (i,j) = \mathit{mbot}\ (i,j)$
using *assms* **by** (*simp add*: *bot-matrix-def*)
qed

lemma *restrict-disjoint-right*:
assumes *disjoint ls ns*
shows $ms\langle ks\langle f\rangle ls\rangle ns = \mathit{mbot}$
proof (*rule ext*, *rule prod-cases*)
fix *i j*
have $(ms\langle ks\langle f\rangle ls\rangle ns)\ (i,j) = (\mathit{if}\ i \in \mathit{set}\ ms \land j \in \mathit{set}\ ns\ \mathit{then}\ \mathit{if}\ i \in \mathit{set}\ ks \land j \in \mathit{set}\ ls\ \mathit{then}\ f\ (i,j)\ \mathit{else}\ \mathit{bot}\ \mathit{else}\ \mathit{bot})$

by (*simp add*: *restrict-matrix-def*)
thus $(ms\langle ks\langle f\rangle ls\rangle ns)\ (i,j) = mbot\ (i,j)$
using *assms* **by** (*simp add*: *bot-matrix-def*)
qed

The following lemma expresses the equality of a matrix and a product of two matrices componentwise.

lemma *restrict-nonempty-product-eq*:
fixes $f\ g\ h$ :: $('a::finite,'b::idempotent\text{-}semiring)\ square$
assumes $\neg\ k \in set\ ks$
and $\neg\ l \in set\ ls$
and $\neg\ m \in set\ ms$
shows $(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms) = (k\#ks)\langle h\rangle(m\#ms) \longleftrightarrow [k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle[m] = [k]\langle h\rangle[m] \wedge [k]\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms = [k]\langle h\rangle ms \wedge ks\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle[m] = ks\langle h\rangle[m] \wedge ks\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms = ks\langle h\rangle ms$
proof –
have *1*: *disjoint* $[k]$ *ks* $\wedge$ *disjoint* $[m]$ *ms*
by (*auto simp add*: *assms*(*1*,*3*))
have *2*: $[k]\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle[m] = [k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle[m]$
proof –
have $[k]\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle[m] = [k]\langle([k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle[m]) \oplus ([k]\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms) \oplus (ks\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle[m]) \oplus (ks\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms)\rangle[m]$
by (*metis assms*(*2*) *restrict-nonempty-product*)
also have ... $= [k]\langle[k]\langle f\rangle[l] \odot [l]\langle g\rangle[m]\rangle[m] \oplus [k]\langle[k]\langle f\rangle ls \odot ls\langle g\rangle[m]\rangle[m] \oplus [k]\langle[k]\langle f\rangle[l] \odot [l]\langle g\rangle ms\rangle[m] \oplus [k]\langle[k]\langle f\rangle ls \odot ls\langle g\rangle ms\rangle[m] \oplus [k]\langle ks\langle f\rangle[l] \odot [l]\langle g\rangle[m]\rangle[m] \oplus [k]\langle ks\langle f\rangle ls \odot ls\langle g\rangle[m]\rangle[m] \oplus [k]\langle ks\langle f\rangle[l] \odot [l]\langle g\rangle ms\rangle[m] \oplus [k]\langle ks\langle f\rangle ls \odot ls\langle g\rangle ms\rangle[m]$
by (*simp add*: *matrix-bounded-semilattice-sup-bot.sup-monoid.add-assoc restrict-sup*)
also have ... $= [k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle[m] \oplus [k]\langle[k]\langle[k]\langle f\rangle[l] \odot [l]\langle g\rangle ms\rangle ms\rangle[m] \oplus [k]\langle[k]\langle[k]\langle f\rangle ls \odot ls\langle g\rangle ms\rangle ms\rangle[m] \oplus [k]\langle ks\langle ks\langle f\rangle[l] \odot [l]\langle g\rangle[m]\rangle[m]\rangle[m] \oplus [k]\langle ks\langle ks\langle f\rangle ls \odot ls\langle g\rangle[m]\rangle[m]\rangle[m] \oplus [k]\langle ks\langle ks\langle f\rangle[l] \odot [l]\langle g\rangle ms\rangle ms\rangle[m] \oplus [k]\langle ks\langle ks\langle f\rangle ls \odot ls\langle g\rangle ms\rangle ms\rangle[m]$
by (*simp add*: *restrict-times*)
also have ... $= [k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle[m]$
using *1* **by** (*metis restrict-disjoint-left restrict-disjoint-right matrix-bounded-semilattice-sup-bot.sup-monoid.add-0-right*)
finally show *?thesis*
.
qed
have *3*: $[k]\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle ms = [k]\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms$
proof –
have $[k]\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle ms = [k]\langle([k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle[m]) \oplus ([k]\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms) \oplus (ks\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle[m]) \oplus (ks\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms)\rangle ms$

by (*metis assms*(*2*) *restrict-nonempty-product*)
also have ... $= [k]\langle[k]\langle f\rangle[l] \odot [l]\langle g\rangle[m]\rangle ms \oplus [k]\langle[k]\langle f\rangle ls \odot ls\langle g\rangle[m]\rangle ms \oplus [k]\langle[k]\langle f\rangle[l] \odot [l]\langle g\rangle ms\rangle ms \oplus [k]\langle[k]\langle f\rangle ls \odot ls\langle g\rangle ms\rangle ms \oplus [k]\langle ks\langle f\rangle[l] \odot [l]\langle g\rangle[m]\rangle ms \oplus [k]\langle ks\langle f\rangle ls \odot ls\langle g\rangle[m]\rangle ms \oplus [k]\langle ks\langle f\rangle[l] \odot [l]\langle g\rangle ms\rangle ms \oplus [k]\langle ks\langle f\rangle ls \odot ls\langle g\rangle ms\rangle ms$
by (*simp add*: *matrix-bounded-semilattice-sup-bot.sup-monoid.add-assoc restrict-sup*)
also have ... $= [k]\langle[k]\langle[k]\langle f\rangle[l] \odot [l]\langle g\rangle[m]\rangle[m]\rangle ms \oplus [k]\langle[k]\langle[k]\langle f\rangle ls \odot ls\langle g\rangle[m]\rangle[m]\rangle ms \oplus [k]\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms \oplus [k]\langle ks\langle ks\langle f\rangle[l] \odot [l]\langle g\rangle[m]\rangle[m]\rangle ms \oplus [k]\langle ks\langle ks\langle f\rangle ls \odot ls\langle g\rangle[m]\rangle[m]\rangle ms \oplus [k]\langle ks\langle ks\langle f\rangle[l] \odot [l]\langle g\rangle ms\rangle ms\rangle ms \oplus [k]\langle ks\langle ks\langle f\rangle ls \odot ls\langle g\rangle ms\rangle ms\rangle ms$
by (*simp add*: *restrict-times*)
also have ... $= [k]\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms$
using *1* **by** (*metis restrict-disjoint-left restrict-disjoint-right matrix-bounded-semilattice-sup-bot.sup-monoid.add-0-right matrix-bounded-semilattice-sup-bot.sup-monoid.add-0-left*)
finally show *?thesis*
.
qed
have *4*: $ks\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle[m] = ks\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle[m]$
proof −
have $ks\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle[m] = ks\langle([k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle[m]) \oplus ([k]\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms) \oplus (ks\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle[m]) \oplus (ks\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms)\rangle[m]$
by (*metis assms*(*2*) *restrict-nonempty-product*)
also have ... $= ks\langle[k]\langle f\rangle[l] \odot [l]\langle g\rangle[m]\rangle[m] \oplus ks\langle[k]\langle f\rangle ls \odot ls\langle g\rangle[m]\rangle[m] \oplus ks\langle[k]\langle f\rangle[l] \odot [l]\langle g\rangle ms\rangle[m] \oplus ks\langle[k]\langle f\rangle ls \odot ls\langle g\rangle ms\rangle[m] \oplus ks\langle ks\langle f\rangle[l] \odot [l]\langle g\rangle[m]\rangle[m] \oplus ks\langle ks\langle f\rangle ls \odot ls\langle g\rangle[m]\rangle[m] \oplus ks\langle ks\langle f\rangle[l] \odot [l]\langle g\rangle ms\rangle[m] \oplus ks\langle ks\langle f\rangle ls \odot ls\langle g\rangle ms\rangle[m]$
by (*simp add*: *matrix-bounded-semilattice-sup-bot.sup-monoid.add-assoc restrict-sup*)
also have ... $= ks\langle[k]\langle[k]\langle f\rangle[l] \odot [l]\langle g\rangle[m]\rangle[m]\rangle[m] \oplus ks\langle[k]\langle[k]\langle f\rangle ls \odot ls\langle g\rangle[m]\rangle[m]\rangle[m] \oplus ks\langle[k]\langle[k]\langle f\rangle[l] \odot [l]\langle g\rangle ms\rangle ms\rangle[m] \oplus ks\langle[k]\langle[k]\langle f\rangle ls \odot ls\langle g\rangle ms\rangle ms\rangle[m] \oplus ks\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle[m] \oplus ks\langle ks\langle ks\langle f\rangle[l] \odot [l]\langle g\rangle ms\rangle ms\rangle[m] \oplus ks\langle ks\langle ks\langle f\rangle ls \odot ls\langle g\rangle ms\rangle ms\rangle[m]$
by (*simp add*: *restrict-times*)
also have ... $= ks\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle[m]$
using *1* **by** (*metis restrict-disjoint-left restrict-disjoint-right matrix-bounded-semilattice-sup-bot.sup-monoid.add-0-right matrix-bounded-semilattice-sup-bot.sup-monoid.add-0-left*)
finally show *?thesis*
.
qed
have *5*: $ks\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle ms = ks\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms$
proof −
have $ks\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle ms = ks\langle([k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle[m]) \oplus ([k]\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms) \oplus (ks\langle f\rangle[l] \odot$

$[l]\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle[m]) \oplus (ks\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms))\rangle ms$
by (*metis assms*(*2*) *restrict-nonempty-product*)
also have ... $= ks\langle[k]\langle f\rangle[l] \odot [l]\langle g\rangle[m]\rangle ms \oplus ks\langle[k]\langle f\rangle ls \odot ls\langle g\rangle[m]\rangle ms \oplus ks\langle[k]\langle f\rangle[l] \odot [l]\langle g\rangle ms\rangle ms \oplus ks\langle[k]\langle f\rangle ls \odot ls\langle g\rangle ms\rangle ms \oplus ks\langle ks\langle f\rangle[l] \odot [l]\langle g\rangle[m]\rangle ms \oplus ks\langle ks\langle f\rangle ls \odot ls\langle g\rangle[m]\rangle ms \oplus ks\langle ks\langle f\rangle[l] \odot [l]\langle g\rangle ms\rangle ms \oplus ks\langle ks\langle f\rangle ls \odot ls\langle g\rangle ms\rangle ms$
by (*simp add*: *matrix-bounded-semilattice-sup-bot.sup-monoid.add-assoc restrict-sup*)
also have ... $= ks\langle[k]\langle[k]\langle f\rangle[l] \odot [l]\langle g\rangle[m]\rangle[m]\rangle ms \oplus ks\langle[k]\langle[k]\langle f\rangle ls \odot ls\langle g\rangle[m]\rangle[m]\rangle ms \oplus ks\langle[k]\langle[k]\langle f\rangle[l] \odot [l]\langle g\rangle ms\rangle ms\rangle ms \oplus ks\langle[k]\langle[k]\langle f\rangle ls \odot ls\langle g\rangle ms\rangle ms\rangle ms \oplus ks\langle ks\langle ks\langle f\rangle[l] \odot [l]\langle g\rangle[m]\rangle[m]\rangle ms \oplus ks\langle ks\langle ks\langle f\rangle ls \odot ls\langle g\rangle[m]\rangle[m]\rangle ms \oplus ks\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms$
by (*simp add*: *restrict-times*)
also have ... $= ks\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms$
using *1* **by** (*metis restrict-disjoint-left restrict-disjoint-right matrix-bounded-semilattice-sup-bot.sup-monoid.add-0-left*)
finally show *?thesis*
.
qed
have $(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms) = (k\#ks)\langle h\rangle(m\#ms) \longleftrightarrow (k\#ks)\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle(m\#ms) = (k\#ks)\langle h\rangle(m\#ms)$
by (*simp add*: *restrict-times*)
also have ... $\longleftrightarrow [k]\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle[m] = [k]\langle h\rangle[m] \wedge [k]\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle ms = [k]\langle h\rangle ms \wedge ks\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle[m] = ks\langle h\rangle[m] \wedge ks\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle ms = ks\langle h\rangle ms$
by (*meson restrict-nonempty-eq*)
also have ... $\longleftrightarrow [k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle[m] = [k]\langle h\rangle[m] \wedge [k]\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms = [k]\langle h\rangle ms \wedge ks\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle[m] = ks\langle h\rangle[m] \wedge ks\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms = ks\langle h\rangle ms$
using *2 3 4 5* **by** *simp*
finally show *?thesis*
by *simp*
qed

The following lemma gives a componentwise characterisation of the inequality of a matrix and a product of two matrices.

lemma *restrict-nonempty-product-less-eq*:
fixes *f g h* :: (*'a::finite,'b::idempotent-semiring*) *square*
assumes $\neg\ k \in set\ ks$
and $\neg\ l \in set\ ls$
and $\neg\ m \in set\ ms$
shows $(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms) \preceq (k\#ks)\langle h\rangle(m\#ms) \longleftrightarrow [k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle[m] \preceq [k]\langle h\rangle[m] \wedge [k]\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms \preceq [k]\langle h\rangle ms \wedge ks\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle[m] \preceq ks\langle h\rangle[m] \wedge ks\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms \preceq ks\langle h\rangle ms$
proof –
have *1*: $[k]\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle[m] = [k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle[m]$
by (*metis assms restrict-nonempty-product-eq restrict-times*)

have *2*: $[k]\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle ms = [k]\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms$
by (*metis assms restrict-nonempty-product-eq restrict-times*)
have *3*: $ks\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle[m] = ks\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle[m]$
by (*metis assms restrict-nonempty-product-eq restrict-times*)
have *4*: $ks\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle ms = ks\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms$
by (*metis assms restrict-nonempty-product-eq restrict-times*)
have $(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms) \preceq (k\#ks)\langle h\rangle(m\#ms) \longleftrightarrow (k\#ks)\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle(m\#ms) \preceq (k\#ks)\langle h\rangle(m\#ms)$
by (*simp add*: *restrict-times*)
also have ... $\longleftrightarrow [k]\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle[m] \preceq [k]\langle h\rangle[m] \wedge [k]\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle ms \preceq [k]\langle h\rangle ms \wedge ks\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle[m] \preceq ks\langle h\rangle[m] \wedge ks\langle(k\#ks)\langle f\rangle(l\#ls) \odot (l\#ls)\langle g\rangle(m\#ms)\rangle ms \preceq ks\langle h\rangle ms$
by (*meson restrict-nonempty-less-eq*)
also have ... $\longleftrightarrow [k]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle[m] \preceq [k]\langle h\rangle[m] \wedge [k]\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus [k]\langle f\rangle ls \odot ls\langle g\rangle ms \preceq [k]\langle h\rangle ms \wedge ks\langle f\rangle[l] \odot [l]\langle g\rangle[m] \oplus ks\langle f\rangle ls \odot ls\langle g\rangle[m] \preceq ks\langle h\rangle[m] \wedge ks\langle f\rangle[l] \odot [l]\langle g\rangle ms \oplus ks\langle f\rangle ls \odot ls\langle g\rangle ms \preceq ks\langle h\rangle ms$
using *1 2 3 4* **by** *simp*
finally show *?thesis*
by *simp*
qed

The Kleene star induction laws hold for matrices with a single entry on the diagonal. The matrix g can actually contain a whole row/colum at the appropriate index.

lemma *restrict-star-left-induct*:
fixes *f g* :: (*'a*::*finite*,*'b*::*kleene-algebra*) *square*
shows *distinct ms* $\Longrightarrow [l]\langle f\rangle[l] \odot [l]\langle g\rangle ms \preceq [l]\langle g\rangle ms \Longrightarrow [l]\langle star\ o\ f\rangle[l] \odot [l]\langle g\rangle ms \preceq [l]\langle g\rangle ms$
proof (*induct ms*)
case *Nil* **thus** *?case*
by (*simp add*: *restrict-empty-right*)
next
case (*Cons m ms*)
assume *1*: *distinct ms* $\Longrightarrow [l]\langle f\rangle[l] \odot [l]\langle g\rangle ms \preceq [l]\langle g\rangle ms \Longrightarrow [l]\langle star\ o\ f\rangle[l] \odot [l]\langle g\rangle ms \preceq [l]\langle g\rangle ms$
assume *2*: *distinct* $(m\#ms)$
assume *3*: $[l]\langle f\rangle[l] \odot [l]\langle g\rangle(m\#ms) \preceq [l]\langle g\rangle(m\#ms)$
have *4*: $[l]\langle f\rangle[l] \odot [l]\langle g\rangle[m] \preceq [l]\langle g\rangle[m] \wedge [l]\langle f\rangle[l] \odot [l]\langle g\rangle ms \preceq [l]\langle g\rangle ms$
using *2 3*
by (*metis distinct.simps*(*2*) *distinct-singleton matrix-semilattice-sup.le-sup-iff restrict-nonempty-product-less-eq*)
hence *5*: $[l]\langle star\ o\ f\rangle[l] \odot [l]\langle g\rangle ms \preceq [l]\langle g\rangle ms$
using *1 2* **by** *simp*
have $f\ (l,l) * g\ (l,m) \le g\ (l,m)$
using *4* **by** (*metis restrict-singleton-product restrict-singleton*

less-eq-matrix-def)
  hence *6*: $(f\ (l,l))^\star * g\ (l,m) \le g\ (l,m)$
    by (*simp add*: *star-left-induct-mult*)
  have $[l]\langle star\ o\ f\rangle[l] \odot [l]\langle g\rangle[m] \preceq [l]\langle g\rangle[m]$
  proof (*unfold less-eq-matrix-def*, *rule allI*, *rule prod-cases*)
    fix *i j*
    have $([l]\langle star\ o\ f\rangle[l] \odot [l]\langle g\rangle[m])\ (i,j) = (\bigsqcup_k\ ([l]\langle star\ o\ f\rangle[l])\ (i,k) * ([l]\langle g\rangle[m])\ (k,j))$
      by (*simp add*: *times-matrix-def*)
    also have ... $= (\bigsqcup_k\ (if\ i = l \wedge k = l\ then\ (f\ (i,k))^\star\ else\ bot) * (if\ k = l \wedge j = m\ then\ g\ (k,j)\ else\ bot))$
      by (*simp add*: *restrict-singleton o-def*)
    also have ... $= (\bigsqcup_k\ if\ k = l\ then\ (if\ i = l\ then\ (f\ (i,k))^\star\ else\ bot) * (if\ j = m\ then\ g\ (k,j)\ else\ bot)\ else\ bot)$
      by (*rule sup-monoid.sum.cong*) *auto*
    also have ... $= (if\ i = l\ then\ (f\ (i,l))^\star\ else\ bot) * (if\ j = m\ then\ g\ (l,j)\ else\ bot)$
      by *simp*
    also have ... $= (if\ i = l \wedge j = m\ then\ (f\ (l,l))^\star * g\ (l,m)\ else\ bot)$
      by *simp*
    also have ... $\le ([l]\langle g\rangle[m])\ (i,j)$
      using *6* **by** (*simp add*: *restrict-singleton*)
    finally show $([l]\langle star\ o\ f\rangle[l] \odot [l]\langle g\rangle[m])\ (i,j) \le ([l]\langle g\rangle[m])\ (i,j)$
      .
  qed
  thus $[l]\langle star\ o\ f\rangle[l] \odot [l]\langle g\rangle(m\#ms) \preceq [l]\langle g\rangle(m\#ms)$
    using *2 5* **by** (*metis* (*no-types*, *opaque-lifting*) *matrix-idempotent-semiring.mult-left-dist-sup matrix-semilattice-sup.sup.mono restrict-nonempty-right*)
qed

lemma *restrict-star-right-induct*:
  fixes *f g* :: (*'a*::*finite*,*'b*::*kleene-algebra*) *square*
  shows *distinct ms* $\Longrightarrow ms\langle g\rangle[l] \odot [l]\langle f\rangle[l] \preceq ms\langle g\rangle[l] \Longrightarrow ms\langle g\rangle[l] \odot [l]\langle star\ o\ f\rangle[l] \preceq ms\langle g\rangle[l]$
proof (*induct ms*)
  case *Nil* **thus** *?case*
    by (*simp add*: *restrict-empty-left*)
next
  case (*Cons m ms*)
  assume *1*: *distinct ms* $\Longrightarrow ms\langle g\rangle[l] \odot [l]\langle f\rangle[l] \preceq ms\langle g\rangle[l] \Longrightarrow ms\langle g\rangle[l] \odot [l]\langle star\ o\ f\rangle[l] \preceq ms\langle g\rangle[l]$
  assume *2*: *distinct* (*m#ms*)
  assume *3*: $(m\#ms)\langle g\rangle[l] \odot [l]\langle f\rangle[l] \preceq (m\#ms)\langle g\rangle[l]$
  have *4*: $[m]\langle g\rangle[l] \odot [l]\langle f\rangle[l] \preceq [m]\langle g\rangle[l] \wedge ms\langle g\rangle[l] \odot [l]\langle f\rangle[l] \preceq ms\langle g\rangle[l]$
    using *2 3*
    by (*metis distinct.simps*(*2*) *distinct-singleton matrix-semilattice-sup.le-sup-iff restrict-nonempty-product-less-eq*)
  hence *5*: $ms\langle g\rangle[l] \odot [l]\langle star\ o\ f\rangle[l] \preceq ms\langle g\rangle[l]$

using *1 2* **by** *simp*
have *g* (*m*,*l*) ∗ *f* (*l*,*l*) ≤ *g* (*m*,*l*)
using *4* **by** (*metis restrict-singleton-product restrict-singleton less-eq-matrix-def*)
hence *6*: *g* (*m*,*l*) ∗ (*f* (*l*,*l*))* ≤ *g* (*m*,*l*)
by (*simp add*: *star-right-induct-mult*)
have [*m*]⟨*g*⟩[*l*] ⊙ [*l*]⟨*star o f*⟩[*l*] ⪯ [*m*]⟨*g*⟩[*l*]
proof (*unfold less-eq-matrix-def*, *rule allI*, *rule prod-cases*)
fix *i j*
have ([*m*]⟨*g*⟩[*l*] ⊙ [*l*]⟨*star o f*⟩[*l*]) (*i*,*j*) = ($\bigsqcup_k$ ([*m*]⟨*g*⟩[*l*]) (*i*,*k*) ∗ ([*l*]⟨*star o f*⟩[*l*]) (*k*,*j*))
by (*simp add*: *times-matrix-def*)
also have ... = ($\bigsqcup_k$ (*if i* = *m* ∧ *k* = *l then g* (*i*,*k*) *else bot*) ∗ (*if k* = *l* ∧ *j* = *l then* (*f* (*k*,*j*))* *else bot*))
by (*simp add*: *restrict-singleton o-def*)
also have ... = ($\bigsqcup_k$ *if k* = *l then* (*if i* = *m then g* (*i*,*k*) *else bot*) ∗ (*if j* = *l then* (*f* (*k*,*j*))* *else bot*) *else bot*)
by (*rule sup-monoid.sum.cong*) *auto*
also have ... = (*if i* = *m then g* (*i*,*l*) *else bot*) ∗ (*if j* = *l then* (*f* (*l*,*j*))* *else bot*)
by *simp*
also have ... = (*if i* = *m* ∧ *j* = *l then g* (*m*,*l*) ∗ (*f* (*l*,*l*))* *else bot*)
by *simp*
also have ... ≤ ([*m*]⟨*g*⟩[*l*]) (*i*,*j*)
using *6* **by** (*simp add*: *restrict-singleton*)
finally show ([*m*]⟨*g*⟩[*l*] ⊙ [*l*]⟨*star o f*⟩[*l*]) (*i*,*j*) ≤ ([*m*]⟨*g*⟩[*l*]) (*i*,*j*)
.
qed
thus (*m*#*ms*)⟨*g*⟩[*l*] ⊙ [*l*]⟨*star o f*⟩[*l*] ⪯ (*m*#*ms*)⟨*g*⟩[*l*]
using *2 5*
by (*metis matrix-idempotent-semiring.mult-right-dist-sup matrix-idempotent-semiring.semiring.add-mono restrict-nonempty-left*)
qed

lemma *restrict-pp*:
fixes *f* :: (′*a*,′*b*::*p-algebra*) *square*
shows *ks*⟨⊖⊖*f*⟩*ls* = ⊖⊖(*ks*⟨*f*⟩*ls*)
by (*unfold restrict-matrix-def uminus-matrix-def*) *auto*

lemma *pp-star-commute*:
fixes *f* :: (′*a*,′*b*::*stone-kleene-relation-algebra*) *square*
shows ⊖⊖(*star o f*) = *star o* ⊖⊖*f*
by (*simp add*: *uminus-matrix-def o-def pp-dist-star*)

6.2 Matrices form a Kleene Algebra

Matrices over Kleene algebras form a Kleene algebra using Conway's construction. It remains to prove one unfold and two induction axioms of the Kleene star. Each proof is by induction over the size of the matrix represented by an index list.

interpretation *matrix-kleene-algebra*: *kleene-algebra-var* **where** *sup* = *sup-matrix* **and** *less-eq* = *less-eq-matrix* **and** *less* = *less-matrix* **and** *bot* = *bot-matrix*::('*a*::*enum*,'*b*::*kleene-algebra*) *square* **and** *one* = *one-matrix* **and** *times* = *times-matrix* **and** *star* = *star-matrix*
proof
  fix *y* :: ('*a*,'*b*) *square*
  let *?e* = *enum-class.enum*::'*a list*
  let *?o* = *mone* :: ('*a*,'*b*) *square*
  have ∀ *g* :: ('*a*,'*b*) *square* . *distinct ?e* ⟶ (*?e*⟨*?o*⟩*?e* ⊕ *?e*⟨*g*⟩*?e* ⊙ *star-matrix*' *?e g*) = (*star-matrix*' *?e g*)
  proof (*induct rule*: *list.induct*)
    case *Nil* **thus** *?case*
      by (*simp add*: *restrict-empty-left*)
  next
    case (*Cons k s*)
    let *?t* = *k*#*s*
    assume *1*: ∀ *g* :: ('*a*,'*b*) *square* . *distinct s* ⟶ (*s*⟨*?o*⟩*s* ⊕ *s*⟨*g*⟩*s* ⊙ *star-matrix*' *s g*) = (*star-matrix*' *s g*)
    show ∀ *g* :: ('*a*,'*b*) *square* . *distinct ?t* ⟶ (*?t*⟨*?o*⟩*?t* ⊕ *?t*⟨*g*⟩*?t* ⊙ *star-matrix*' *?t g*) = (*star-matrix*' *?t g*)
    proof (*rule allI*, *rule impI*)
      fix *g* :: ('*a*,'*b*) *square*
      assume *2*: *distinct ?t*
      let *?r* = [*k*]
      let *?a* = *?r*⟨*g*⟩*?r*
      let *?b* = *?r*⟨*g*⟩*s*
      let *?c* = *s*⟨*g*⟩*?r*
      let *?d* = *s*⟨*g*⟩*s*
      let *?as* = *?r*⟨*star o ?a*⟩*?r*
      let *?ds* = *star-matrix*' *s ?d*
      let *?e* = *?a* ⊕ *?b* ⊙ *?ds* ⊙ *?c*
      let *?es* = *?r*⟨*star o ?e*⟩*?r*
      let *?f* = *?d* ⊕ *?c* ⊙ *?as* ⊙ *?b*
      let *?fs* = *star-matrix*' *s ?f*
      have *s*⟨*?ds*⟩*s* = *?ds* ∧ *s*⟨*?fs*⟩*s* = *?fs*
        by (*simp add*: *restrict-star*)
      hence *3*: *?r*⟨*?e*⟩*?r* = *?e* ∧ *s*⟨*?f*⟩*s* = *?f*
        by (*metis* (*no-types*, *lifting*) *restrict-one-left-unit restrict-sup restrict-times*)
      have *4*: *disjoint s ?r* ∧ *disjoint ?r s*
        using *2* **by** *simp*
      hence *5*: *?t*⟨*?o*⟩*?t* = *?r*⟨*?o*⟩*?r* ⊕ *s*⟨*?o*⟩*s*
        by (*auto intro*: *restrict-one*)
      have *6*: *?t*⟨*g*⟩*?t* ⊙ *?es* = *?a* ⊙ *?es* ⊕ *?c* ⊙ *?es*
      proof −
        have *?t*⟨*g*⟩*?t* ⊙ *?es* = (*?a* ⊕ *?b* ⊕ *?c* ⊕ *?d*) ⊙ *?es*
          by (*metis restrict-nonempty*)
        also have ... = *?a* ⊙ *?es* ⊕ *?b* ⊙ *?es* ⊕ *?c* ⊙ *?es* ⊕ *?d* ⊙ *?es*
          by (*simp add*: *matrix-idempotent-semiring.mult-right-dist-sup*)

also have ... = *?a* ⊙ *?es* ⊕ *?c* ⊙ *?es*
using *4* **by** (*simp add*: *times-disjoint*)
finally show *?thesis*
.
qed
have *7*: *?t⟨g⟩?t* ⊙ *?as* ⊙ *?b* ⊙ *?fs* = *?a* ⊙ *?as* ⊙ *?b* ⊙ *?fs* ⊕ *?c* ⊙ *?as* ⊙ *?b* ⊙ *?fs*
proof −
have *?t⟨g⟩?t* ⊙ *?as* ⊙ *?b* ⊙ *?fs* = (*?a* ⊕ *?b* ⊕ *?c* ⊕ *?d*) ⊙ *?as* ⊙ *?b* ⊙ *?fs*
by (*metis restrict-nonempty*)
also have ... = *?a* ⊙ *?as* ⊙ *?b* ⊙ *?fs* ⊕ *?b* ⊙ *?as* ⊙ *?b* ⊙ *?fs* ⊕ *?c* ⊙ *?as* ⊙ *?b* ⊙ *?fs* ⊕ *?d* ⊙ *?as* ⊙ *?b* ⊙ *?fs*
by (*simp add*: *matrix-idempotent-semiring.mult-right-dist-sup*)
also have ... = *?a* ⊙ *?as* ⊙ *?b* ⊙ *?fs* ⊕ *?c* ⊙ *?as* ⊙ *?b* ⊙ *?fs*
using *4* **by** (*simp add*: *times-disjoint*)
finally show *?thesis*
.
qed
have *8*: *?t⟨g⟩?t* ⊙ *?ds* ⊙ *?c* ⊙ *?es* = *?b* ⊙ *?ds* ⊙ *?c* ⊙ *?es* ⊕ *?d* ⊙ *?ds* ⊙ *?c* ⊙ *?es*
proof −
have *?t⟨g⟩?t* ⊙ *?ds* ⊙ *?c* ⊙ *?es* = (*?a* ⊕ *?b* ⊕ *?c* ⊕ *?d*) ⊙ *?ds* ⊙ *?c* ⊙ *?es*
by (*metis restrict-nonempty*)
also have ... = *?a* ⊙ *?ds* ⊙ *?c* ⊙ *?es* ⊕ *?b* ⊙ *?ds* ⊙ *?c* ⊙ *?es* ⊕ *?c* ⊙ *?ds* ⊙ *?c* ⊙ *?es* ⊕ *?d* ⊙ *?ds* ⊙ *?c* ⊙ *?es*
by (*simp add*: *matrix-idempotent-semiring.mult-right-dist-sup*)
also have ... = *?b* ⊙ *?ds* ⊙ *?c* ⊙ *?es* ⊕ *?d* ⊙ *?ds* ⊙ *?c* ⊙ *?es*
using *4* **by** (*metis* (*no-types*, *lifting*) *times-disjoint matrix-idempotent-semiring.mult-left-zero restrict-star matrix-bounded-semilattice-sup-bot.sup-monoid.add-0-right matrix-bounded-semilattice-sup-bot.sup-monoid.add-0-left*)
finally show *?thesis*
.
qed
have *9*: *?t⟨g⟩?t* ⊙ *?fs* = *?b* ⊙ *?fs* ⊕ *?d* ⊙ *?fs*
proof −
have *?t⟨g⟩?t* ⊙ *?fs* = (*?a* ⊕ *?b* ⊕ *?c* ⊕ *?d*) ⊙ *?fs*
by (*metis restrict-nonempty*)
also have ... = *?a* ⊙ *?fs* ⊕ *?b* ⊙ *?fs* ⊕ *?c* ⊙ *?fs* ⊕ *?d* ⊙ *?fs*
by (*simp add*: *matrix-idempotent-semiring.mult-right-dist-sup*)
also have ... = *?b* ⊙ *?fs* ⊕ *?d* ⊙ *?fs*
using *4* **by** (*metis* (*no-types*, *lifting*) *times-disjoint restrict-star matrix-bounded-semilattice-sup-bot.sup-monoid.add-0-right matrix-bounded-semilattice-sup-bot.sup-monoid.add-0-left*)
finally show *?thesis*
.
qed
have *?t⟨?o⟩?t* ⊕ *?t⟨g⟩?t* ⊙ *star-matrix′ ?t g* = *?t⟨?o⟩?t* ⊕ *?t⟨g⟩?t* ⊙ (*?es* ⊕ *?as* ⊙ *?b* ⊙ *?fs* ⊕ *?ds* ⊙ *?c* ⊙ *?es* ⊕ *?fs*)

by (*metis star-matrix'.simps*(*2*))
also have ... = $?t\langle ?o\rangle ?t \oplus ?t\langle g\rangle ?t \odot ?es \oplus ?t\langle g\rangle ?t \odot ?as \odot ?b \odot ?fs \oplus ?t\langle g\rangle ?t \odot ?ds \odot ?c \odot ?es \oplus ?t\langle g\rangle ?t \odot ?fs$
by (*simp add*: *matrix-idempotent-semiring.mult-left-dist-sup matrix-monoid.mult-assoc matrix-semilattice-sup.sup-assoc*)
also have ... = $?r\langle ?o\rangle ?r \oplus s\langle ?o\rangle s \oplus ?a \odot ?es \oplus ?c \odot ?es \oplus ?a \odot ?as \odot ?b \odot ?fs \oplus ?c \odot ?as \odot ?b \odot ?fs \oplus ?b \odot ?ds \odot ?c \odot ?es \oplus ?d \odot ?ds \odot ?c \odot ?es \oplus ?b \odot ?fs \oplus ?d \odot ?fs$
using *5 6 7 8 9* **by** (*simp add*: *matrix-semilattice-sup.sup.assoc*)
also have ... = $(?r\langle ?o\rangle ?r \oplus (?a \odot ?es \oplus ?b \odot ?ds \odot ?c \odot ?es)) \oplus (?b \odot ?fs \oplus ?a \odot ?as \odot ?b \odot ?fs) \oplus (?c \odot ?es \oplus ?d \odot ?ds \odot ?c \odot ?es) \oplus (s\langle ?o\rangle s \oplus (?d \odot ?fs \oplus ?c \odot ?as \odot ?b \odot ?fs))$
by (*simp only*: *matrix-semilattice-sup.sup-assoc matrix-semilattice-sup.sup-commute matrix-semilattice-sup.sup-left-commute*)
also have ... = $(?r\langle ?o\rangle ?r \oplus (?a \odot ?es \oplus ?b \odot ?ds \odot ?c \odot ?es)) \oplus (?r\langle ?o\rangle ?r \odot ?b \odot ?fs \oplus ?a \odot ?as \odot ?b \odot ?fs) \oplus (s\langle ?o\rangle s \odot ?c \odot ?es \oplus ?d \odot ?ds \odot ?c \odot ?es) \oplus (s\langle ?o\rangle s \oplus (?d \odot ?fs \oplus ?c \odot ?as \odot ?b \odot ?fs))$
by (*simp add*: *restrict-one-left-unit*)
also have ... = $(?r\langle ?o\rangle ?r \oplus ?e \odot ?es) \oplus ((?r\langle ?o\rangle ?r \oplus ?a \odot ?as) \odot ?b \odot ?fs) \oplus ((s\langle ?o\rangle s \oplus ?d \odot ?ds) \odot ?c \odot ?es) \oplus (s\langle ?o\rangle s \oplus ?f \odot ?fs)$
by (*simp add*: *matrix-idempotent-semiring.mult-right-dist-sup*)
also have ... = $(?r\langle ?o\rangle ?r \oplus ?e \odot ?es) \oplus ((?r\langle ?o\rangle ?r \oplus ?a \odot ?as) \odot ?b \odot ?fs) \oplus ((s\langle ?o\rangle s \oplus ?d \odot ?ds) \odot ?c \odot ?es) \oplus ?fs$
using *1 2 3* **by** (*metis distinct.simps*(*2*))
also have ... = $(?r\langle ?o\rangle ?r \oplus ?e \odot ?es) \oplus ((?r\langle ?o\rangle ?r \oplus ?a \odot ?as) \odot ?b \odot ?fs) \oplus (?ds \odot ?c \odot ?es) \oplus ?fs$
using *1 2*
by (*metis distinct.simps*(*2*) *restrict-one-left-unit restrict-times*)
also have ... = $?es \oplus ((?r\langle ?o\rangle ?r \oplus ?a \odot ?as) \odot ?b \odot ?fs) \oplus (?ds \odot ?c \odot ?es) \oplus ?fs$
using *3* **by** (*metis restrict-star-unfold*)
also have ... = $?es \oplus ?as \odot ?b \odot ?fs \oplus ?ds \odot ?c \odot ?es \oplus ?fs$
by (*metis* (*no-types*, *lifting*) *restrict-one-left-unit restrict-star-unfold restrict-times*)
also have ... = *star-matrix' ?t g*
by (*metis star-matrix'.simps*(*2*))
finally show $?t\langle ?o\rangle ?t \oplus ?t\langle g\rangle ?t \odot$ *star-matrix' ?t g* = *star-matrix' ?t g*
.
qed
qed
thus $?o \oplus y \odot y^{\odot} \preceq y^{\odot}$
by (*simp add*: *enum-distinct restrict-all*)
next
fix *x y z* :: ('*a*,'*b*) *square*
let *?e* = *enum-class.enum*::'*a list*
have $\forall g\ h$ :: ('*a*,'*b*) *square* . $\forall zs$. *distinct ?e* $\wedge$ *distinct zs* $\longrightarrow$ $(?e\langle g\rangle ?e \odot ?e\langle h\rangle zs \preceq ?e\langle h\rangle zs \longrightarrow$ *star-matrix' ?e g* $\odot\ ?e\langle h\rangle zs \preceq ?e\langle h\rangle zs)$
proof (*induct rule*: *list.induct*)
case *Nil* **thus** *?case*

by (*simp add*: *restrict-empty-left*)
case (*Cons k s*)
let $?t = k\#s$
assume 1: $\forall\, g\ h :: ('a,'b)\ square\ .\ \forall\, zs\ .\ distinct\ s \wedge distinct\ zs \longrightarrow (s\langle g\rangle s \odot s\langle h\rangle zs \preceq s\langle h\rangle zs \longrightarrow star\text{-}matrix'\ s\ g \odot s\langle h\rangle zs \preceq s\langle h\rangle zs)$
show $\forall\, g\ h :: ('a,'b)\ square\ .\ \forall\, zs\ .\ distinct\ ?t \wedge distinct\ zs \longrightarrow (?t\langle g\rangle ?t \odot ?t\langle h\rangle zs \preceq ?t\langle h\rangle zs \longrightarrow star\text{-}matrix'\ ?t\ g \odot ?t\langle h\rangle zs \preceq ?t\langle h\rangle zs)$
proof (*intro allI*)
fix $g\ h :: ('a,'b)\ square$
fix $zs :: 'a\ list$
show $distinct\ ?t \wedge distinct\ zs \longrightarrow (?t\langle g\rangle ?t \odot ?t\langle h\rangle zs \preceq ?t\langle h\rangle zs \longrightarrow star\text{-}matrix'\ ?t\ g \odot ?t\langle h\rangle zs \preceq ?t\langle h\rangle zs)$
proof (*cases zs*)
case *Nil* **thus** *?thesis*
by (*metis restrict-empty-right restrict-star restrict-times*)
next
case (*Cons y ys*)
assume 2: $zs = y\#ys$
show $distinct\ ?t \wedge distinct\ zs \longrightarrow (?t\langle g\rangle ?t \odot ?t\langle h\rangle zs \preceq ?t\langle h\rangle zs \longrightarrow star\text{-}matrix'\ ?t\ g \odot ?t\langle h\rangle zs \preceq ?t\langle h\rangle zs)$
proof (*intro impI*)
let $?y = [y]$
assume 3: $distinct\ ?t \wedge distinct\ zs$
hence 4: $distinct\ s \wedge distinct\ ys \wedge \neg\ k \in set\ s \wedge \neg\ y \in set\ ys$
using 2 **by** *simp*
let $?r = [k]$
let $?a = ?r\langle g\rangle ?r$
let $?b = ?r\langle g\rangle s$
let $?c = s\langle g\rangle ?r$
let $?d = s\langle g\rangle s$
let $?as = ?r\langle star\ o\ ?a\rangle ?r$
let $?ds = star\text{-}matrix'\ s\ ?d$
let $?e = ?a \oplus ?b \odot ?ds \odot ?c$
let $?es = ?r\langle star\ o\ ?e\rangle ?r$
let $?f = ?d \oplus ?c \odot ?as \odot ?b$
let $?fs = star\text{-}matrix'\ s\ ?f$
let $?ha = ?r\langle h\rangle ?y$
let $?hb = ?r\langle h\rangle ys$
let $?hc = s\langle h\rangle ?y$
let $?hd = s\langle h\rangle ys$
assume $?t\langle g\rangle ?t \odot ?t\langle h\rangle zs \preceq ?t\langle h\rangle zs$
hence 5: $?a \odot ?ha \oplus ?b \odot ?hc \preceq ?ha \wedge ?a \odot ?hb \oplus ?b \odot ?hd \preceq ?hb \wedge ?c \odot ?ha \oplus ?d \odot ?hc \preceq ?hc \wedge ?c \odot ?hb \oplus ?d \odot ?hd \preceq ?hd$
using $2\ 3\ 4$ **by** *simp*
(*meson matrix-semilattice-sup.le-sup-iff restrict-nonempty-product-less-eq*)
have 6: $s\langle ?ds\rangle s = ?ds \wedge s\langle ?fs\rangle s = ?fs$
by (*simp add*: *restrict-star*)
hence 7: $?r\langle ?e\rangle ?r = ?e \wedge s\langle ?f\rangle s = ?f$

by (*metis* (*no-types*, *lifting*) *restrict-one-left-unit restrict-sup restrict-times*)
have *8*: *disjoint s ?r* $\wedge$ *disjoint ?r s*
using *3* **by** *simp*
have *9*: *?es* $\odot$ *?t*$\langle h \rangle$*zs* = *?es* $\odot$ *?ha* $\oplus$ *?es* $\odot$ *?hb*
proof –
have *?es* $\odot$ *?t*$\langle h \rangle$*zs* = *?es* $\odot$ (*?ha* $\oplus$ *?hb* $\oplus$ *?hc* $\oplus$ *?hd*)
using *2* **by** (*metis restrict-nonempty*)
also have ... = *?es* $\odot$ *?ha* $\oplus$ *?es* $\odot$ *?hb* $\oplus$ *?es* $\odot$ *?hc* $\oplus$ *?es* $\odot$ *?hd*
by (*simp add*: *matrix-idempotent-semiring.mult-left-dist-sup*)
also have ... = *?es* $\odot$ *?ha* $\oplus$ *?es* $\odot$ *?hb*
using *8* **by** (*simp add*: *times-disjoint*)
finally show *?thesis*
.
qed
have *10*: *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?t*$\langle h \rangle$*zs* = *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?hc* $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?hd*
proof –
have *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?t*$\langle h \rangle$*zs* = *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ (*?ha* $\oplus$ *?hb* $\oplus$ *?hc* $\oplus$ *?hd*)
using *2* **by** (*metis restrict-nonempty*)
also have ... = *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?ha* $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?hb* $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?hc* $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?hd*
by (*simp add*: *matrix-idempotent-semiring.mult-left-dist-sup*)
also have ... = *?as* $\odot$ *?b* $\odot$ (*?fs* $\odot$ *?ha*) $\oplus$ *?as* $\odot$ *?b* $\odot$ (*?fs* $\odot$ *?hb*) $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?hc* $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?hd*
by (*simp add*: *matrix-monoid.mult-assoc*)
also have ... = *?as* $\odot$ *?b* $\odot$ *mbot* $\oplus$ *?as* $\odot$ *?b* $\odot$ *mbot* $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?hc* $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?hd*
using *6 8* **by** (*metis* (*no-types*) *times-disjoint*)
also have ... = *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?hc* $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?hd*
by *simp*
finally show *?thesis*
.
qed
have *11*: *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?t*$\langle h \rangle$*zs* = *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?ha* $\oplus$ *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?hb*
proof –
have *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?t*$\langle h \rangle$*zs* = *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ (*?ha* $\oplus$ *?hb* $\oplus$ *?hc* $\oplus$ *?hd*)
using *2* **by** (*metis restrict-nonempty*)
also have ... = *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?ha* $\oplus$ *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?hb* $\oplus$ *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?hc* $\oplus$ *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?hd*
by (*simp add*: *matrix-idempotent-semiring.mult-left-dist-sup*)
also have ... = *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?ha* $\oplus$ *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?hb* $\oplus$ *?ds* $\odot$ *?c* $\odot$ (*?es* $\odot$ *?hc*) $\oplus$ *?ds* $\odot$ *?c* $\odot$ (*?es* $\odot$ *?hd*)
by (*simp add*: *matrix-monoid.mult-assoc*)
also have ... = *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?ha* $\oplus$ *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?hb* $\oplus$ *?ds* $\odot$ *?c* $\odot$ *mbot* $\oplus$ *?ds* $\odot$ *?c* $\odot$ *mbot*

using *8* **by** (*metis times-disjoint*)
also have ... = *?ds* ⊙ *?c* ⊙ *?es* ⊙ *?ha* ⊕ *?ds* ⊙ *?c* ⊙ *?es* ⊙ *?hb*
by *simp*
finally show *?thesis*
.
qed
have *12*: *?fs* ⊙ *?t*⟨*h*⟩*zs* = *?fs* ⊙ *?hc* ⊕ *?fs* ⊙ *?hd*
proof −
have *?fs* ⊙ *?t*⟨*h*⟩*zs* = *?fs* ⊙ (*?ha* ⊕ *?hb* ⊕ *?hc* ⊕ *?hd*)
using *2* **by** (*metis restrict-nonempty*)
also have ... = *?fs* ⊙ *?ha* ⊕ *?fs* ⊙ *?hb* ⊕ *?fs* ⊙ *?hc* ⊕ *?fs* ⊙ *?hd*
by (*simp add*: *matrix-idempotent-semiring.mult-left-dist-sup*)
also have ... = *?fs* ⊙ *?hc* ⊕ *?fs* ⊙ *?hd*
using *6 8* **by** (*metis* (*no-types*) *times-disjoint*
matrix-bounded-semilattice-sup-bot.sup-monoid.add-0-left)
finally show *?thesis*
.
qed
have *13*: *?es* ⊙ *?ha* ≼ *?ha*
proof −
have *?b* ⊙ *?ds* ⊙ *?c* ⊙ *?ha* ≼ *?b* ⊙ *?ds* ⊙ *?hc*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-isotone*
matrix-monoid.mult-assoc)
also have ... ≼ *?b* ⊙ *?hc*
using *1 3 5* **by** (*simp add*:
matrix-idempotent-semiring.mult-right-isotone matrix-monoid.mult-assoc
restrict-sublist)
also have ... ≼ *?ha*
using *5* **by** *simp*
finally have *?e* ⊙ *?ha* ≼ *?ha*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-dist-sup*)
thus *?thesis*
using *7* **by** (*simp add*: *restrict-star-left-induct*)
qed
have *14*: *?es* ⊙ *?hb* ≼ *?hb*
proof −
have *?b* ⊙ *?ds* ⊙ *?c* ⊙ *?hb* ≼ *?b* ⊙ *?ds* ⊙ *?hd*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-isotone*
matrix-monoid.mult-assoc)
also have ... ≼ *?b* ⊙ *?hd*
using *1 4 5* **by** (*simp add*:
matrix-idempotent-semiring.mult-right-isotone matrix-monoid.mult-assoc
restrict-sublist)
also have ... ≼ *?hb*
using *5* **by** *simp*
finally have *?e* ⊙ *?hb* ≼ *?hb*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-dist-sup*)
thus *?thesis*
using *4 7* **by** (*simp add*: *restrict-star-left-induct*)

qed
have *15*: *?fs* ⊙ *?hc* ≼ *?hc*
proof –
have *?c* ⊙ *?as* ⊙ *?b* ⊙ *?hc* ≼ *?c* ⊙ *?as* ⊙ *?ha*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-isotone matrix-monoid.mult-assoc*)
also have ... ≼ *?c* ⊙ *?ha*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-isotone matrix-monoid.mult-assoc restrict-star-left-induct restrict-sublist*)
also have ... ≼ *?hc*
using *5* **by** *simp*
finally have *?f* ⊙ *?hc* ≼ *?hc*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-dist-sup*)
thus *?thesis*
using *1 3 7* **by** *simp*
qed
have *16*: *?fs* ⊙ *?hd* ≼ *?hd*
proof –
have *?c* ⊙ *?as* ⊙ *?b* ⊙ *?hd* ≼ *?c* ⊙ *?as* ⊙ *?hb*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-isotone matrix-monoid.mult-assoc*)
also have ... ≼ *?c* ⊙ *?hb*
using *4 5* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-isotone matrix-monoid.mult-assoc restrict-star-left-induct restrict-sublist*)
also have ... ≼ *?hd*
using *5* **by** *simp*
finally have *?f* ⊙ *?hd* ≼ *?hd*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-dist-sup*)
thus *?thesis*
using *1 4 7* **by** *simp*
qed
have *17*: *?as* ⊙ *?b* ⊙ *?fs* ⊙ *?hc* ≼ *?ha*
proof –
have *?as* ⊙ *?b* ⊙ *?fs* ⊙ *?hc* ≼ *?as* ⊙ *?b* ⊙ *?hc*
using *15* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-isotone matrix-monoid.mult-assoc*)
also have ... ≼ *?as* ⊙ *?ha*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-isotone matrix-monoid.mult-assoc*)
also have ... ≼ *?ha*
using *5* **by** (*simp add*: *restrict-star-left-induct restrict-sublist*)
finally show *?thesis*
.
qed
have *18*: *?as* ⊙ *?b* ⊙ *?fs* ⊙ *?hd* ≼ *?hb*
proof –
have *?as* ⊙ *?b* ⊙ *?fs* ⊙ *?hd* ≼ *?as* ⊙ *?b* ⊙ *?hd*
using *16* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-isotone matrix-monoid.mult-assoc*)

also have ... $\preceq$ *?as* $\odot$ *?hb*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-isotone matrix-monoid.mult-assoc*)
also have ... $\preceq$ *?hb*
using *4 5* **by** (*simp add*: *restrict-star-left-induct restrict-sublist*)
finally show *?thesis*
.
qed
have *19*: *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?ha* $\preceq$ *?hc*
proof −
have *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?ha* $\preceq$ *?ds* $\odot$ *?c* $\odot$ *?ha*
using *13* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-isotone matrix-monoid.mult-assoc*)
also have ... $\preceq$ *?ds* $\odot$ *?hc*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-isotone matrix-monoid.mult-assoc*)
also have ... $\preceq$ *?hc*
using *1 3 5* **by** (*simp add*: *restrict-sublist*)
finally show *?thesis*
.
qed
have *20*: *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?hb* $\preceq$ *?hd*
proof −
have *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?hb* $\preceq$ *?ds* $\odot$ *?c* $\odot$ *?hb*
using *14* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-isotone matrix-monoid.mult-assoc*)
also have ... $\preceq$ *?ds* $\odot$ *?hd*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-right-isotone matrix-monoid.mult-assoc*)
also have ... $\preceq$ *?hd*
using *1 4 5* **by** (*simp add*: *restrict-sublist*)
finally show *?thesis*
.
qed
have *21*: *?es* $\odot$ *?ha* $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?hc* $\preceq$ *?ha*
using *13 17 matrix-semilattice-sup.le-supI* **by** *blast*
have *22*: *?es* $\odot$ *?hb* $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?hd* $\preceq$ *?hb*
using *14 18 matrix-semilattice-sup.le-supI* **by** *blast*
have *23*: *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?ha* $\oplus$ *?fs* $\odot$ *?hc* $\preceq$ *?hc*
using *15 19 matrix-semilattice-sup.le-supI* **by** *blast*
have *24*: *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?hb* $\oplus$ *?fs* $\odot$ *?hd* $\preceq$ *?hd*
using *16 20 matrix-semilattice-sup.le-supI* **by** *blast*
have *star-matrix′ ?t g* $\odot$ *?t*$\langle h \rangle$*zs* = (*?es* $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\oplus$ *?ds* $\odot$ *?c* $\odot$ *?es* $\oplus$ *?fs*) $\odot$ *?t*$\langle h \rangle$*zs*
by (*metis star-matrix′.simps*(*2*))
also have ... = *?es* $\odot$ *?t*$\langle h \rangle$*zs* $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?t*$\langle h \rangle$*zs* $\oplus$ *?ds* $\odot$ *?c* $\odot$ *?es* $\odot$ *?t*$\langle h \rangle$*zs* $\oplus$ *?fs* $\odot$ *?t*$\langle h \rangle$*zs*
by (*simp add*: *matrix-idempotent-semiring.mult-right-dist-sup*)
also have ... = *?es* $\odot$ *?ha* $\oplus$ *?es* $\odot$ *?hb* $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\odot$ *?hc* $\oplus$ *?as* $\odot$

$?b \odot ?fs \odot ?hd \oplus ?ds \odot ?c \odot ?es \odot ?ha \oplus ?ds \odot ?c \odot ?es \odot ?hb \oplus ?fs \odot ?hc \oplus ?fs \odot ?hd$

using *9 10 11 12* **by** (*simp only*: *matrix-semilattice-sup.sup-assoc*)

also have ... $= (?es \odot ?ha \oplus ?as \odot ?b \odot ?fs \odot ?hc) \oplus (?es \odot ?hb \oplus ?as \odot ?b \odot ?fs \odot ?hd) \oplus (?ds \odot ?c \odot ?es \odot ?ha \oplus ?fs \odot ?hc) \oplus (?ds \odot ?c \odot ?es \odot ?hb \oplus ?fs \odot ?hd)$

by (*simp only*: *matrix-semilattice-sup.sup-assoc matrix-semilattice-sup.sup-commute matrix-semilattice-sup.sup-left-commute*)

also have ... $\preceq ?ha \oplus ?hb \oplus ?hc \oplus ?hd$

using *21 22 23 24 matrix-semilattice-sup.sup.mono* **by** *blast*

also have ... $= ?t\langle h\rangle zs$

using *2* **by** (*metis restrict-nonempty*)

finally show $star\text{-}matrix'\ ?t\ g \odot ?t\langle h\rangle zs \preceq ?t\langle h\rangle zs$

.

qed

qed

qed

qed

hence $\forall zs .\ distinct\ zs \longrightarrow (y \odot ?e\langle x\rangle zs \preceq ?e\langle x\rangle zs \longrightarrow y^{\odot} \odot ?e\langle x\rangle zs \preceq ?e\langle x\rangle zs)$

by (*simp add*: *enum-distinct restrict-all*)

thus $y \odot x \preceq x \longrightarrow y^{\odot} \odot x \preceq x$

by (*metis restrict-all enum-distinct*)

next

fix $x\ y\ z :: ('a,'b)\ square$

let $?e = enum\text{-}class.enum::'a\ list$

have $\forall g\ h :: ('a,'b)\ square .\ \forall zs .\ distinct\ ?e \wedge distinct\ zs \longrightarrow (zs\langle h\rangle ?e \odot ?e\langle g\rangle ?e \preceq zs\langle h\rangle ?e \longrightarrow zs\langle h\rangle ?e \odot star\text{-}matrix'\ ?e\ g \preceq zs\langle h\rangle ?e)$

proof (*induct rule*:*list.induct*)

case *Nil* **thus** *?case*

by (*simp add*: *restrict-empty-left*)

case (*Cons k s*)

let $?t = k\#s$

assume *1*: $\forall g\ h :: ('a,'b)\ square .\ \forall zs .\ distinct\ s \wedge distinct\ zs \longrightarrow (zs\langle h\rangle s \odot s\langle g\rangle s \preceq zs\langle h\rangle s \longrightarrow zs\langle h\rangle s \odot star\text{-}matrix'\ s\ g \preceq zs\langle h\rangle s)$

show $\forall g\ h :: ('a,'b)\ square .\ \forall zs .\ distinct\ ?t \wedge distinct\ zs \longrightarrow (zs\langle h\rangle ?t \odot ?t\langle g\rangle ?t \preceq zs\langle h\rangle ?t \longrightarrow zs\langle h\rangle ?t \odot star\text{-}matrix'\ ?t\ g \preceq zs\langle h\rangle ?t)$

proof (*intro allI*)

fix $g\ h :: ('a,'b)\ square$

fix $zs :: 'a\ list$

show $distinct\ ?t \wedge distinct\ zs \longrightarrow (zs\langle h\rangle ?t \odot ?t\langle g\rangle ?t \preceq zs\langle h\rangle ?t \longrightarrow zs\langle h\rangle ?t \odot star\text{-}matrix'\ ?t\ g \preceq zs\langle h\rangle ?t)$

proof (*cases zs*)

case *Nil* **thus** *?thesis*

by (*metis restrict-empty-left restrict-star restrict-times*)

next

case (*Cons y ys*)

assume *2*: $zs = y\#ys$

show $distinct\ ?t \wedge distinct\ zs \longrightarrow (zs\langle h\rangle ?t \odot ?t\langle g\rangle ?t \preceq zs\langle h\rangle ?t \longrightarrow$

$zs\langle h\rangle ?t \odot$ *star-matrix'* $?t\ g \preceq zs\langle h\rangle ?t$)

proof (*intro impI*)

let $?y = [y]$

assume *3*: *distinct* $?t \wedge$ *distinct zs*

hence *4*: *distinct* $s \wedge$ *distinct ys* $\wedge \neg\ k \in$ *set* $s \wedge \neg\ y \in$ *set ys*

using *2* **by** *simp*

let $?r = [k]$

let $?a = ?r\langle g\rangle ?r$

let $?b = ?r\langle g\rangle s$

let $?c = s\langle g\rangle ?r$

let $?d = s\langle g\rangle s$

let $?as = ?r\langle star\ o\ ?a\rangle ?r$

let $?ds =$ *star-matrix'* $s\ ?d$

let $?e = ?a \oplus ?b \odot ?ds \odot ?c$

let $?es = ?r\langle star\ o\ ?e\rangle ?r$

let $?f = ?d \oplus ?c \odot ?as \odot ?b$

let $?fs =$ *star-matrix'* $s\ ?f$

let $?ha = ?y\langle h\rangle ?r$

let $?hb = ?y\langle h\rangle s$

let $?hc = ys\langle h\rangle ?r$

let $?hd = ys\langle h\rangle s$

assume $zs\langle h\rangle ?t \odot ?t\langle g\rangle ?t \preceq zs\langle h\rangle ?t$

hence *5*: $?ha \odot ?a \oplus ?hb \odot ?c \preceq ?ha \wedge ?ha \odot ?b \oplus ?hb \odot ?d \preceq ?hb \wedge$
$?hc \odot ?a \oplus ?hd \odot ?c \preceq ?hc \wedge ?hc \odot ?b \oplus ?hd \odot ?d \preceq ?hd$

using *2 3 4*

using *restrict-nonempty-product-less-eq* **by** *blast*

have *6*: $s\langle ?ds\rangle s = ?ds \wedge s\langle ?fs\rangle s = ?fs$

by (*simp add*: *restrict-star*)

hence *7*: $?r\langle ?e\rangle ?r = ?e \wedge s\langle ?f\rangle s = ?f$

by (*metis* (*no-types*, *lifting*) *restrict-one-left-unit restrict-sup*
restrict-times)

have *8*: *disjoint* $s\ ?r \wedge$ *disjoint* $?r\ s$

using *3* **by** *simp*

have *9*: $zs\langle h\rangle ?t \odot ?es = ?ha \odot ?es \oplus ?hc \odot ?es$

proof −

have $zs\langle h\rangle ?t \odot ?es = (?ha \oplus ?hb \oplus ?hc \oplus ?hd) \odot ?es$

using *2* **by** (*metis restrict-nonempty*)

also have ... $= ?ha \odot ?es \oplus ?hb \odot ?es \oplus ?hc \odot ?es \oplus ?hd \odot ?es$

by (*simp add*: *matrix-idempotent-semiring.mult-right-dist-sup*)

also have ... $= ?ha \odot ?es \oplus ?hc \odot ?es$

using *8* **by** (*simp add*: *times-disjoint*)

finally show *?thesis*

.

qed

have *10*: $zs\langle h\rangle ?t \odot ?as \odot ?b \odot ?fs = ?ha \odot ?as \odot ?b \odot ?fs \oplus ?hc \odot$
$?as \odot ?b \odot ?fs$

proof −

have $zs\langle h\rangle ?t \odot ?as \odot ?b \odot ?fs = (?ha \oplus ?hb \oplus ?hc \oplus ?hd) \odot ?as \odot$
$?b \odot ?fs$

using *2* **by** (*metis restrict-nonempty*)
also have ... = *?ha* ⊙ *?as* ⊙ *?b* ⊙ *?fs* ⊕ *?hb* ⊙ *?as* ⊙ *?b* ⊙ *?fs* ⊕ *?hc* ⊙ *?as* ⊙ *?b* ⊙ *?fs* ⊕ *?hd* ⊙ *?as* ⊙ *?b* ⊙ *?fs*
by (*simp add*: *matrix-idempotent-semiring.mult-right-dist-sup*)
also have ... = *?ha* ⊙ *?as* ⊙ *?b* ⊙ *?fs* ⊕ *mbot* ⊙ *?b* ⊙ *?fs* ⊕ *?hc* ⊙ *?as* ⊙ *?b* ⊙ *?fs* ⊕ *mbot* ⊙ *?b* ⊙ *?fs*
using *8* **by** (*metis* (*no-types*) *times-disjoint*)
also have ... = *?ha* ⊙ *?as* ⊙ *?b* ⊙ *?fs* ⊕ *?hc* ⊙ *?as* ⊙ *?b* ⊙ *?fs*
by *simp*
finally show *?thesis*
.
qed
have *11*: *zs*⟨*h*⟩*?t* ⊙ *?ds* ⊙ *?c* ⊙ *?es* = *?hb* ⊙ *?ds* ⊙ *?c* ⊙ *?es* ⊕ *?hd* ⊙ *?ds* ⊙ *?c* ⊙ *?es*
proof −
have *zs*⟨*h*⟩*?t* ⊙ *?ds* ⊙ *?c* ⊙ *?es* = (*?ha* ⊕ *?hb* ⊕ *?hc* ⊕ *?hd*) ⊙ *?ds* ⊙ *?c* ⊙ *?es*
using *2* **by** (*metis restrict-nonempty*)
also have ... = *?ha* ⊙ *?ds* ⊙ *?c* ⊙ *?es* ⊕ *?hb* ⊙ *?ds* ⊙ *?c* ⊙ *?es* ⊕ *?hc* ⊙ *?ds* ⊙ *?c* ⊙ *?es* ⊕ *?hd* ⊙ *?ds* ⊙ *?c* ⊙ *?es*
by (*simp add*: *matrix-idempotent-semiring.mult-right-dist-sup*)
also have ... = *mbot* ⊙ *?c* ⊙ *?es* ⊕ *?hb* ⊙ *?ds* ⊙ *?c* ⊙ *?es* ⊕ *mbot* ⊙ *?c* ⊙ *?es* ⊕ *?hd* ⊙ *?ds* ⊙ *?c* ⊙ *?es*
using *6 8* **by** (*metis* (*no-types*) *times-disjoint*)
also have ... = *?hb* ⊙ *?ds* ⊙ *?c* ⊙ *?es* ⊕ *?hd* ⊙ *?ds* ⊙ *?c* ⊙ *?es*
by *simp*
finally show *?thesis*
.
qed
have *12*: *zs*⟨*h*⟩*?t* ⊙ *?fs* = *?hb* ⊙ *?fs* ⊕ *?hd* ⊙ *?fs*
proof −
have *zs*⟨*h*⟩*?t* ⊙ *?fs* = (*?ha* ⊕ *?hb* ⊕ *?hc* ⊕ *?hd*) ⊙ *?fs*
using *2* **by** (*metis restrict-nonempty*)
also have ... = *?ha* ⊙ *?fs* ⊕ *?hb* ⊙ *?fs* ⊕ *?hc* ⊙ *?fs* ⊕ *?hd* ⊙ *?fs*
by (*simp add*: *matrix-idempotent-semiring.mult-right-dist-sup*)
also have ... = *?hb* ⊙ *?fs* ⊕ *?hd* ⊙ *?fs*
using *6 8* **by** (*metis* (*no-types*) *times-disjoint matrix-bounded-semilattice-sup-bot.sup-monoid.add-0-right matrix-bounded-semilattice-sup-bot.sup-monoid.add-0-left*)
finally show *?thesis*
.
qed
have *13*: *?ha* ⊙ *?es* ⪯ *?ha*
proof −
have *?ha* ⊙ *?b* ⊙ *?ds* ⊙ *?c* ⪯ *?hb* ⊙ *?ds* ⊙ *?c*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-isotone*)
also have ... ⪯ *?hb* ⊙ *?c*
using *1 4 5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-isotone restrict-sublist*)

also have ... $\preceq$ *?ha*
using *5* **by** *simp*
finally have *?ha* $\odot$ *?e* $\preceq$ *?ha*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-dist-sup matrix-monoid.mult-assoc*)
thus *?thesis*
using *7* **by** (*simp add*: *restrict-star-right-induct*)
qed
have *14*: *?hb* $\odot$ *?fs* $\preceq$ *?hb*
proof −
have *?hb* $\odot$ *?c* $\odot$ *?as* $\odot$ *?b* $\preceq$ *?ha* $\odot$ *?as* $\odot$ *?b*
using *5* **by** (*metis matrix-semilattice-sup.le-supE matrix-idempotent-semiring.mult-left-isotone*)
also have ... $\preceq$ *?ha* $\odot$ *?b*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-isotone restrict-star-right-induct restrict-sublist*)
also have ... $\preceq$ *?hb*
using *5* **by** *simp*
finally have *?hb* $\odot$ *?f* $\preceq$ *?hb*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-dist-sup matrix-monoid.mult-assoc*)
thus *?thesis*
using *1 3 7* **by** *simp*
qed
have *15*: *?hc* $\odot$ *?es* $\preceq$ *?hc*
proof −
have *?hc* $\odot$ *?b* $\odot$ *?ds* $\odot$ *?c* $\preceq$ *?hd* $\odot$ *?ds* $\odot$ *?c*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-isotone*)
also have ... $\preceq$ *?hd* $\odot$ *?c*
using *1 4 5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-isotone restrict-sublist*)
also have ... $\preceq$ *?hc*
using *5* **by** *simp*
finally have *?hc* $\odot$ *?e* $\preceq$ *?hc*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-dist-sup matrix-monoid.mult-assoc*)
thus *?thesis*
using *4 7* **by** (*simp add*: *restrict-star-right-induct*)
qed
have *16*: *?hd* $\odot$ *?fs* $\preceq$ *?hd*
proof −
have *?hd* $\odot$ *?c* $\odot$ *?as* $\odot$ *?b* $\preceq$ *?hc* $\odot$ *?as* $\odot$ *?b*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-isotone*)
also have ... $\preceq$ *?hc* $\odot$ *?b*
using *4 5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-isotone restrict-star-right-induct restrict-sublist*)
also have ... $\preceq$ *?hd*
using *5* **by** *simp*
finally have *?hd* $\odot$ *?f* $\preceq$ *?hd*

using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-dist-sup matrix-monoid.mult-assoc*)
thus *?thesis*
using *1 4 7* **by** *simp*
qed
have *17*: *?hb* ⊙ *?ds* ⊙ *?c* ⊙ *?es* ≼ *?ha*
proof −
have *?hb* ⊙ *?ds* ⊙ *?c* ⊙ *?es* ≼ *?hb* ⊙ *?c* ⊙ *?es*
using *1 4 5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-isotone restrict-sublist*)
also have ... ≼ *?ha* ⊙ *?es*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-isotone*)
also have ... ≼ *?ha*
using *13* **by** *simp*
finally show *?thesis*
.
qed
have *18*: *?ha* ⊙ *?as* ⊙ *?b* ⊙ *?fs* ≼ *?hb*
proof −
have *?ha* ⊙ *?as* ⊙ *?b* ⊙ *?fs* ≼ *?ha* ⊙ *?b* ⊙ *?fs*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-isotone restrict-star-right-induct restrict-sublist*)
also have ... ≼ *?hb* ⊙ *?fs*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-isotone*)
also have ... ≼ *?hb*
using *14* **by** *simp*
finally show *?thesis*
by *simp*
qed
have *19*: *?hd* ⊙ *?ds* ⊙ *?c* ⊙ *?es* ≼ *?hc*
proof −
have *?hd* ⊙ *?ds* ⊙ *?c* ⊙ *?es* ≼ *?hd* ⊙ *?c* ⊙ *?es*
using *1 4 5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-isotone restrict-sublist*)
also have ... ≼ *?hc* ⊙ *?es*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-isotone*)
also have ... ≼ *?hc*
using *15* **by** *simp*
finally show *?thesis*
by *simp*
qed
have *20*: *?hc* ⊙ *?as* ⊙ *?b* ⊙ *?fs* ≼ *?hd*
proof −
have *?hc* ⊙ *?as* ⊙ *?b* ⊙ *?fs* ≼ *?hc* ⊙ *?b* ⊙ *?fs*
using *4 5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-isotone restrict-star-right-induct restrict-sublist*)
also have ... ≼ *?hd* ⊙ *?fs*
using *5* **by** (*simp add*: *matrix-idempotent-semiring.mult-left-isotone*)
also have ... ≼ *?hd*

using *16* **by** *simp*
finally show *?thesis*
by *simp*
qed
have *21*: $?ha \odot ?es \oplus ?hb \odot ?ds \odot ?c \odot ?es \preceq ?ha$
using *13 17 matrix-semilattice-sup.le-supI* **by** *blast*
have *22*: $?ha \odot ?as \odot ?b \odot ?fs \oplus ?hb \odot ?fs \preceq ?hb$
using *14 18 matrix-semilattice-sup.le-supI* **by** *blast*
have *23*: $?hc \odot ?es \oplus ?hd \odot ?ds \odot ?c \odot ?es \preceq ?hc$
using *15 19 matrix-semilattice-sup.le-supI* **by** *blast*
have *24*: $?hc \odot ?as \odot ?b \odot ?fs \oplus ?hd \odot ?fs \preceq ?hd$
using *16 20 matrix-semilattice-sup.le-supI* **by** *blast*
have $zs\langle h\rangle ?t \odot star\text{-}matrix'\ ?t\ g = zs\langle h\rangle ?t \odot (?es \oplus ?as \odot ?b \odot ?fs \oplus ?ds \odot ?c \odot ?es \oplus ?fs)$
by (*metis star-matrix'.simps(2)*)
also have ... $= zs\langle h\rangle ?t \odot ?es \oplus zs\langle h\rangle ?t \odot ?as \odot ?b \odot ?fs \oplus zs\langle h\rangle ?t \odot ?ds \odot ?c \odot ?es \oplus zs\langle h\rangle ?t \odot ?fs$
by (*simp add*: *matrix-idempotent-semiring.mult-left-dist-sup matrix-monoid.mult-assoc*)
also have ... $= ?ha \odot ?es \oplus ?hc \odot ?es \oplus ?ha \odot ?as \odot ?b \odot ?fs \oplus ?hc \odot ?as \odot ?b \odot ?fs \oplus ?hb \odot ?ds \odot ?c \odot ?es \oplus ?hd \odot ?ds \odot ?c \odot ?es \oplus ?hb \odot ?fs \oplus ?hd \odot ?fs$
using *9 10 11 12* **by** (*simp add*: *matrix-semilattice-sup.sup-assoc*)
also have ... $= (?ha \odot ?es \oplus ?hb \odot ?ds \odot ?c \odot ?es) \oplus (?ha \odot ?as \odot ?b \odot ?fs \oplus ?hb \odot ?fs) \oplus (?hc \odot ?es \oplus ?hd \odot ?ds \odot ?c \odot ?es) \oplus (?hc \odot ?as \odot ?b \odot ?fs \oplus ?hd \odot ?fs)$
using *9 10 11 12* **by** (*simp only*: *matrix-semilattice-sup.sup-assoc matrix-semilattice-sup.sup-commute matrix-semilattice-sup.sup-left-commute*)
also have ... $\preceq ?ha \oplus ?hb \oplus ?hc \oplus ?hd$
using *21 22 23 24 matrix-semilattice-sup.sup.mono* **by** *blast*
also have ... $= zs\langle h\rangle ?t$
using *2* **by** (*metis restrict-nonempty*)
finally show $zs\langle h\rangle ?t \odot star\text{-}matrix'\ ?t\ g \preceq zs\langle h\rangle ?t$
.
qed
qed
qed
qed
hence $\forall zs .\ distinct\ zs \longrightarrow (zs\langle x\rangle ?e \odot y \preceq zs\langle x\rangle ?e \longrightarrow zs\langle x\rangle ?e \odot y^{\odot} \preceq zs\langle x\rangle ?e)$
by (*simp add*: *enum-distinct restrict-all*)
thus $x \odot y \preceq x \longrightarrow x \odot y^{\odot} \preceq x$
by (*metis restrict-all enum-distinct*)
qed

6.3 Matrices form a Stone-Kleene Relation Algebra

Matrices over Stone-Kleene relation algebras form a Stone-Kleene relation algebra. It remains to prove the axiom about the interaction of Kleene star and double complement.

interpretation *matrix-stone-kleene-relation-algebra*: *stone-kleene-relation-algebra*
where *sup* = *sup-matrix* **and** *inf* = *inf-matrix* **and** *less-eq* = *less-eq-matrix* **and**
less = *less-matrix* **and** *bot* =
bot-matrix::('*a*::*enum*,'*b*::*stone-kleene-relation-algebra*) *square* **and** *top* =
top-matrix **and** *uminus* = *uminus-matrix* **and** *one* = *one-matrix* **and** *times* =
times-matrix **and** *conv* = *conv-matrix* **and** *star* = *star-matrix*
proof
  fix x :: ('*a*,'*b*) *square*
  let *?e* = *enum-class.enum*::'*a list*
  let *?o* = *mone* :: ('*a*,'*b*) *square*
  show $\ominus\ominus(x^\odot) = (\ominus\ominus x)^\odot$
  proof (*rule matrix-order.order-antisym*)
    have ∀ *g* :: ('*a*,'*b*) *square* . *distinct ?e* ⟶ ⊖⊖(*star-matrix' ?e* (⊖⊖*g*)) =
star-matrix' ?e (⊖⊖*g*)
    proof (*induct rule*: *list.induct*)
      case *Nil* **thus** *?case*
        by *simp*
    next
      case (*Cons k s*)
      let *?t* = *k*#*s*
      assume *1*: ∀ *g* :: ('*a*,'*b*) *square* . *distinct s* ⟶ ⊖⊖(*star-matrix' s* (⊖⊖*g*)) =
star-matrix' s (⊖⊖*g*)
      show ∀ *g* :: ('*a*,'*b*) *square* . *distinct ?t* ⟶ ⊖⊖(*star-matrix' ?t* (⊖⊖*g*)) =
star-matrix' ?t (⊖⊖*g*)
      proof (*rule allI*, *rule impI*)
        fix *g* :: ('*a*,'*b*) *square*
        assume *2*: *distinct ?t*
        let *?r* = [*k*]
        let *?a* = *?r*⟨⊖⊖*g*⟩*?r*
        let *?b* = *?r*⟨⊖⊖*g*⟩*s*
        let *?c* = *s*⟨⊖⊖*g*⟩*?r*
        let *?d* = *s*⟨⊖⊖*g*⟩*s*
        let *?as* = *?r*⟨*star o ?a*⟩*?r*
        let *?ds* = *star-matrix' s ?d*
        let *?e* = *?a* ⊕ *?b* ⊙ *?ds* ⊙ *?c*
        let *?es* = *?r*⟨*star o ?e*⟩*?r*
        let *?f* = *?d* ⊕ *?c* ⊙ *?as* ⊙ *?b*
        let *?fs* = *star-matrix' s ?f*
        have *s*⟨*?ds*⟩*s* = *?ds* ∧ *s*⟨*?fs*⟩*s* = *?fs*
          by (*simp add*: *restrict-star*)
        have *3*: ⊖⊖*?a* = *?a* ∧ ⊖⊖*?b* = *?b* ∧ ⊖⊖*?c* = *?c* ∧ ⊖⊖*?d* = *?d*
          by (*metis matrix-p-algebra.regular-closed-p restrict-pp*)
        hence *4*: ⊖⊖*?as* = *?as*
          by (*metis pp-star-commute restrict-pp*)
        hence ⊖⊖*?f* = *?f*
          using *3* **by** (*metis matrix-stone-algebra.regular-closed-sup*
matrix-stone-relation-algebra.regular-mult-closed)
        hence *5*: ⊖⊖*?fs* = *?fs*

using *1 2* **by** (*metis distinct.simps*(*2*))
have *6*: $\ominus\ominus$*?ds* = *?ds*
using *1 2* **by** (*simp add*: *restrict-pp*)
hence $\ominus\ominus$*?e* = *?e*
using *3* **by** (*metis matrix-stone-algebra.regular-closed-sup matrix-stone-relation-algebra.regular-mult-closed*)
hence *7*: $\ominus\ominus$*?es* = *?es*
by (*metis pp-star-commute restrict-pp*)
have $\ominus\ominus$(*star-matrix′ ?t* ($\ominus\ominus g$)) = $\ominus\ominus$(*?es* $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\oplus$ *?ds* $\odot$ *?c* $\odot$ *?es* $\oplus$ *?fs*)
by (*metis star-matrix′.simps*(*2*))
also have ... = $\ominus\ominus$*?es* $\oplus$ $\ominus\ominus$*?as* $\odot$ $\ominus\ominus$*?b* $\odot$ $\ominus\ominus$*?fs* $\oplus$ $\ominus\ominus$*?ds* $\odot$ $\ominus\ominus$*?c* $\odot$ $\ominus\ominus$*?es* $\oplus$ $\ominus\ominus$*?fs*
by (*simp add*: *matrix-stone-relation-algebra.pp-dist-comp*)
also have ... = *?es* $\oplus$ *?as* $\odot$ *?b* $\odot$ *?fs* $\oplus$ *?ds* $\odot$ *?c* $\odot$ *?es* $\oplus$ *?fs*
using *3 4 5 6 7* **by** *simp*
finally show $\ominus\ominus$(*star-matrix′ ?t* ($\ominus\ominus g$)) = *star-matrix′ ?t* ($\ominus\ominus g$)
by (*metis star-matrix′.simps*(*2*))
qed
qed
hence $(\ominus\ominus x)^{\odot} = \ominus\ominus((\ominus\ominus x)^{\odot})$
by (*simp add*: *enum-distinct restrict-all*)
thus $\ominus\ominus(x^{\odot}) \preceq (\ominus\ominus x)^{\odot}$
by (*metis matrix-kleene-algebra.star.circ-isotone matrix-p-algebra.pp-increasing matrix-p-algebra.pp-isotone*)
next
have *?o* $\oplus \ominus\ominus x \odot \ominus\ominus(x^{\odot}) \preceq \ominus\ominus(x^{\odot})$
by (*metis matrix-kleene-algebra.star-left-unfold-equal matrix-p-algebra.sup-pp-semi-commute matrix-stone-relation-algebra.pp-dist-comp*)
thus $(\ominus\ominus x)^{\odot} \preceq \ominus\ominus(x^{\odot})$
using *matrix-kleene-algebra.star-left-induct* **by** *fastforce*
qed
qed

interpretation *matrix-stone-kleene-relation-algebra-consistent*: *stone-kleene-relation-algebra-consistent* **where** *sup* = *sup-matrix* **and** *inf* = *inf-matrix* **and** *less-eq* = *less-eq-matrix* **and** *less* = *less-matrix* **and** *bot* = *bot-matrix* :: (*′a*::*enum*,*′b*::*stone-kleene-relation-algebra-consistent*) *square* **and** *top* = *top-matrix* **and** *uminus* = *uminus-matrix* **and** *one* = *one-matrix* **and** *times* = *times-matrix* **and** *conv* = *conv-matrix* **and** *star* = *star-matrix*
..

interpretation *matrix-stone-kleene-relation-algebra-tarski*: *stone-kleene-relation-algebra-tarski* **where** *sup* = *sup-matrix* **and** *inf* = *inf-matrix* **and** *less-eq* = *less-eq-matrix* **and** *less* = *less-matrix* **and** *bot* = *bot-matrix* :: (*′a*::*enum*,*′b*::*stone-kleene-relation-algebra-tarski*) *square* **and** *top* = *top-matrix* **and** *uminus* = *uminus-matrix* **and** *one* = *one-matrix* **and** *times* = *times-matrix* **and** *conv* = *conv-matrix* **and** *star* = *star-matrix*
..

interpretation *matrix-stone-kleene-relation-algebra-tarski-consistent*: *stone-kleene-relation-algebra-tarski-consistent* **where** *sup* = *sup-matrix* **and** *inf* = *inf-matrix* **and** *less-eq* = *less-eq-matrix* **and** *less* = *less-matrix* **and** *bot* = *bot-matrix* :: (*'a*::*enum*,*'b*::*stone-kleene-relation-algebra-tarski-consistent*) *square* **and** *top* = *top-matrix* **and** *uminus* = *uminus-matrix* **and** *one* = *one-matrix* **and** *times* = *times-matrix* **and** *conv* = *conv-matrix* **and** *star* = *star-matrix*
..

end

References

[1] A. Armstrong, S. Foster, G. Struth, and T. Weber. Relation algebra. *Archive of Formal Proofs*, 2016, first version 2014.

[2] A. Armstrong, V. B. F. Gomes, G. Struth, and T. Weber. Kleene algebra. *Archive of Formal Proofs*, 2016, first version 2013.

[3] T. Asplund. Formalizing the Kleene star for square matrices. Bachelor Thesis IT 14 002, Uppsala Universitet, Department of Information Technology, 2014.

[4] R. J. R. Back and J. von Wright. Reasoning algebraically about loops. *Acta Inf.*, 36(4):295–334, 1999.

[5] S. L. Bloom and Z. Ésik. *Iteration Theories: The Equational Logic of Iterative Processes.* Springer, 1993.

[6] E. Cohen. Separation and reduction. In R. Backhouse and J. N. Oliveira, editors, *Mathematics of Program Construction*, volume 1837 of *Lecture Notes in Computer Science*, pages 45–59. Springer, 2000.

[7] J. H. Conway. *Regular Algebra and Finite Machines.* Chapman and Hall, 1971.

[8] S. Foster and G. Struth. Regular algebras. *Archive of Formal Proofs*, 2016, first version 2014.

[9] W. Guttmann. Algebras for iteration and infinite computations. *Acta Inf.*, 49(5):343–359, 2012.

[10] W. Guttmann. Relation-algebraic verification of Prim's minimum spanning tree algorithm. In A. Sampaio and F. Wang, editors, *Theoretical Aspects of Computing – ICTAC 2016*, volume 9965 of *Lecture Notes in Computer Science*, pages 51–68. Springer, 2016.

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

[12] W. Guttmann. Stone relation algebras. In P. Höfner, D. Pous, and G. Struth, editors, *Relational and Algebraic Methods in Computer Science*, volume 10226 of *Lecture Notes in Computer Science*, pages 127–143. Springer, 2017.

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

[14] D. Kozen. Typed Kleene algebra. Technical Report TR98-1669, Cornell University, 1998.

[15] B. Möller. Kleene getting lazy. *Sci. Comput. Programming*, 65(2):195–214, 2007.

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

[17] J. von Wright. Towards a refinement algebra. *Sci. Comput. Programming*, 51(1–2):23–45, 2004.