Higher Globular Catoids and Quantales

Cameron Calk and Georg Struth

February 1, 2024

Abstract

We formalise strict 2-catoids, 2-categories, 2-Kleene algebras and 2-quantales, as well as their ω-variants. Due to strictness, the cells of these higher categories have globular shape. We use a single-set approach, generalised to catoids and based on type classes. The higher Kleene algebras and quantales introduced extend features of modal and concurrent Kleene algebras and quantales. They arise for instance as powerset extensions of higher catoids, and have been used in algebraic confluence proofs in higher-dimensional rewriting. Details are described in two companion articles.

Contents

1 Introductory remarks

We extend formalisations of catoids, categories and quantales from the AFP [5, 4, 3] to higher variants, as described in a companion article [2]. The categories, in particular, are formalised in a single-set approach. They are strict so that their cells have globular shape. We formalise the cases of 2 and ω separately. First, strict 2-categories are important in category theory: the category of all small categories, for example, forms such a category. Second, strict ω-categories are simply given by pairs of strict 2-categories in all dimensions, so that many properties for ω generalise easily from 2-properties. Fourth, Isabelle's Nitpick tool can find interesting counterexamples at dimension 2, but not for ω. Finally, in the type classes formalising our ω-structures, the numerical indices of higher operations cannot simply be instantiated to a fixed value such as 2. Applications of higher Kleene algebras and quantales in higher-dimensional rewriting are explained in [1], where these structures were introduced.

With higher catoids, the partial compositions of cells in higher categories are relaxed to multioperations, which assign each pair of elements to a set of elements, so that mapping to the empty set captures partiality. In addition, a composition of two elements may be undefined even though the target of the first equals the source of the second in a given dimension.

2 2-Catoids

theory *Two-Catoid*
 imports *Catoids.Catoid*

begin

We define 2-catoids and in particular (strict) 2-categories as local functional 2-catoids. With Isabelle we first need to make two copies of catoids for the 0-structure and 1-structure.

2.1 0-Structures and 1-structures.

class *multimagma0* =
  fixes *mcomp0* :: $'a \Rightarrow 'a \Rightarrow 'a\ set$ (**infixl** $\odot_0$ *70*)

begin

sublocale *mm0*: *multimagma mcomp0*⟨*proof*⟩

abbreviation $\Delta_0 \equiv mm0.\Delta$

abbreviation *conv0* :: $'a\ set \Rightarrow 'a\ set \Rightarrow 'a\ set$ (**infixl** $*_0$ *70*) **where**
 $X *_0 Y \equiv mm0.conv\ X\ Y$

lemma $X *_0 Y = (\bigcup x \in X. \bigcup y \in Y.\ x \odot_0 y)$
  ⟨*proof*⟩

end

class *multimagma1* =
  fixes *mcomp1* :: $'a \Rightarrow 'a \Rightarrow 'a\ set$ (**infixl** $\odot_1$ *70*)

begin

sublocale *mm1*: *multimagma mcomp1*⟨*proof*⟩

abbreviation $\Delta_1 \equiv mm1.\Delta$

abbreviation *conv1* :: $'a\ set \Rightarrow 'a\ set \Rightarrow 'a\ set$ (**infixl** $*_1$ *70*) **where**
  $X *_1 Y \equiv mm1.conv\ X\ Y$

end

class *multisemigroup0* = *multimagma0* +
  assumes *assoc*: $(\bigcup v \in y \odot_0 z.\ x \odot_0 v) = (\bigcup v \in x \odot_0 y.\ v \odot_0 z)$

sublocale *multisemigroup0* ⊆ *msg0*: *multisemigroup mcomp0*
  ⟨*proof*⟩

class *multisemigroup1* = *multimagma1* +
  assumes *assoc*: $(\bigcup v \in y \odot_1 z.\ x \odot_1 v) = (\bigcup v \in x \odot_1 y.\ v \odot_1 z)$

sublocale *multisemigroup1* ⊆ *msg1*: *multisemigroup mcomp1*

$\langle proof \rangle$

class *st-multimagma0* = *multimagma0* +
fixes σ_0 :: $'a \Rightarrow 'a$
and τ_0 :: $'a \Rightarrow 'a$
assumes *Dst0*: $x \odot_0 y \neq \{\} \Longrightarrow \tau_0\ x = \sigma_0\ y$
and *src0-absorb* [*simp*]: $\sigma_0\ x \odot_0 x = \{x\}$
and *tgt0-absorb* [*simp*]: $x \odot_0 \tau_0\ x = \{x\}$

begin

sublocale *stmm0*: *st-multimagma mcomp0* σ_0 τ_0
$\langle proof \rangle$

abbreviation *s0fix* $\equiv$ *stmm0.sfix*

abbreviation *t0fix* $\equiv$ *stmm0.tfix*

abbreviation Src_0 $\equiv$ *stmm0.Src*

abbreviation Tgt_0 $\equiv$ *stmm0.Tgt*

end

class *st-multimagma1* = *multimagma1* +
fixes σ_1 :: $'a \Rightarrow 'a$
and τ_1 :: $'a \Rightarrow 'a$
assumes *Dst1*: $x \odot_1 y \neq \{\} \Longrightarrow \tau_1\ x = \sigma_1\ y$
and *src1-absorb* [*simp*]: $\sigma_1\ x \odot_1 x = \{x\}$
and *tgt1-absorb* [*simp*]: $x \odot_1 \tau_1\ x = \{x\}$

begin

sublocale *stmm1*: *st-multimagma mcomp1* σ_1 τ_1
$\langle proof \rangle$

abbreviation *s1fix* $\equiv$ *stmm1.sfix*

abbreviation *t1fix* $\equiv$ *stmm1.tfix*

abbreviation Src_1 $\equiv$ *stmm1.Src*

abbreviation Tgt_1 $\equiv$ *stmm1.Tgt*

end

class *catoid0* = *st-multimagma0* + *multisemigroup0*

sublocale *catoid0* $\subseteq$ *stmsg0*: *catoid mcomp0* σ_0 $\tau_0 \langle proof \rangle$

class *catoid1* = *st-multimagma1* + *multisemigroup1*

sublocale *catoid1* $\subseteq$ *stmsg1*: *catoid mcomp1* σ_1 τ_1⟨*proof*⟩

class *local-catoid0* = *catoid0* +
  assumes *src0-local*: Src_0 $(x \odot_0 \sigma_0\ y) \subseteq Src_0$ $(x \odot_0 y)$
  and *tgt0-local*: Tgt_0 $(\tau_0\ x \odot_0 y) \subseteq Tgt_0$ $(x \odot_0 y)$

class *local-catoid1* = *catoid1* +
  assumes *l1-local*: Src_1 $(x \odot_1 \sigma_1\ y) \subseteq Src_1$ $(x \odot_1 y)$
  and *r1-local*: Tgt_1 $(\tau_1\ x \odot_1 y) \subseteq Tgt_1$ $(x \odot_1 y)$

sublocale *local-catoid0* $\subseteq$ *ssmsg0*: *local-catoid mcomp0* σ_0 τ_0
⟨*proof*⟩

sublocale *local-catoid1* $\subseteq$ *stmsg1*: *local-catoid mcomp1* σ_1 τ_1
⟨*proof*⟩

class *functional-magma0* = *multimagma0* +
  assumes *functionality0*: $x \in y \odot_0 z \Longrightarrow x' \in y \odot_0 z \Longrightarrow x = x'$

sublocale *functional-magma0* $\subseteq$ *pm0*: *functional-magma mcomp0*
⟨*proof*⟩

class *functional-magma1* = *multimagma1* +
  assumes *functionality1*: $x \in y \odot_1 z \Longrightarrow x' \in y \odot_1 z \Longrightarrow x = x'$

sublocale *functional-magma1* $\subseteq$ *pm1*: *functional-magma mcomp1*
⟨*proof*⟩

class *functional-semigroup0* = *functional-magma0* + *multisemigroup0*

sublocale *functional-semigroup0* $\subseteq$ *psg0*: *functional-semigroup mcomp0*⟨*proof*⟩

class *functional-semigroup1* = *functional-magma1* + *multisemigroup1*

sublocale *functional-semigroup1* $\subseteq$ *psg1*: *functional-semigroup mcomp1*⟨*proof*⟩

class *functional-catoid0* = *functional-semigroup0* + *catoid0*

sublocale *functional-catoid0* $\subseteq$ *psg0*: *functional-catoid mcomp0* σ_0 τ_0⟨*proof*⟩

class *functional-catoid1* = *functional-semigroup1* + *catoid1*

sublocale *functional-catoid1* $\subseteq$ *psg1*: *functional-catoid mcomp1* σ_1 τ_1⟨*proof*⟩

class *single-set-category0* = *functional-catoid0* + *local-catoid0*

sublocale *single-set-category0* ⊆ *sscat0*: *single-set-category mcomp0* σ_0 τ_0⟨*proof*⟩

class *single-set-category1* = *functional-catoid1* + *local-catoid1*

sublocale *single-set-category1* ⊆ *sscat1*: *single-set-category mcomp1* σ_1 τ_1⟨*proof*⟩

2.2 2-Catoids

We define 2-catoids and 2-categories.

class *two-st-multimagma* = *st-multimagma0* + *st-multimagma1* +
assumes *comm-s0s1*: $\sigma_0\ (\sigma_1\ x) = \sigma_1\ (\sigma_0\ x)$
and *comm-s0t1*: $\sigma_0\ (\tau_1\ x) = \tau_1\ (\sigma_0\ x)$
and *comm-t0s1*: $\tau_0\ (\sigma_1\ x) = \sigma_1\ (\tau_0\ x)$
and *comm-t0t1*: $\tau_0\ (\tau_1\ x) = \tau_1\ (\tau_0\ x)$
assumes *interchange*: $(w \odot_1 x) *_0 (y \odot_1 z) \subseteq (w \odot_0 y) *_1 (x \odot_0 z)$
and *s1-hom*: $Src_1\ (x \odot_0 y) \subseteq \sigma_1\ x \odot_0 \sigma_1\ y$
and *t1-hom*: $Tgt_1\ (x \odot_0 y) \subseteq \tau_1\ x \odot_0 \tau_1\ y$
and *s0-hom*: $Src_0\ (x \odot_1 y) \subseteq \sigma_0\ x \odot_1 \sigma_0\ y$
and *t0-hom*: $Tgt_0\ (x \odot_1 y) \subseteq \tau_0\ x \odot_1 \tau_0\ y$
and *s1s0* [*simp*]: $\sigma_1\ (\sigma_0\ x) = \sigma_0\ x$
and *s1t0* [*simp*]: $\sigma_1\ (\tau_0\ x) = \tau_0\ x$
and *t1s0* [*simp*]: $\tau_1\ (\sigma_0\ x) = \sigma_0\ x$
and *t1t0* [*simp*]: $\tau_1\ (\tau_0\ x) = \tau_0\ x$

class *two-st-multimagma-strong* = *two-st-multimagma* +
assumes *s1-hom-strong*: $Src_1\ (x \odot_0 y) = \sigma_1\ x \odot_0 \sigma_1\ y$
and *t1-hom-strong*: $Tgt_1\ (x \odot_0 y) = \tau_1\ x \odot_0 \tau_1\ y$

context *two-st-multimagma*
begin

sublocale *twolropp*: *two-st-multimagma* $\lambda x\ y.\ y \odot_0 x\ \tau_0\ \sigma_0\ \lambda x\ y.\ y \odot_1 x\ \tau_1\ \sigma_1$
⟨*proof*⟩

lemma *s0s1* [*simp*]: $\sigma_0\ (\sigma_1\ x) = \sigma_0\ x$
⟨*proof*⟩

lemma *s0t1* [*simp*]: $\sigma_0\ (\tau_1\ x) = \sigma_0\ x$
⟨*proof*⟩

lemma *t0s1* [*simp*]: $\tau_0\ (\sigma_1\ x) = \tau_0\ x$
⟨*proof*⟩

lemma *t1t1* [*simp*]: $\tau_0\ (\tau_1\ x) = \tau_0\ x$
⟨*proof*⟩

lemma *src0-comp1*: $\Delta_1\ x\ y \Longrightarrow Src_0\ (x \odot_1 y) = \{\sigma_0\ x\}$
⟨*proof*⟩

lemma *src0-comp1-var*: $\Delta_1\ x\ y \Longrightarrow Src_0\ (x \odot_1\ y) = \{\sigma_0\ y\}$
⟨*proof*⟩

lemma *tgt0-comp1*: $\Delta_1\ x\ y \Longrightarrow Tgt_0\ (x \odot_1\ y) = \{\tau_0\ x\}$
⟨*proof*⟩

lemma *tgt0-comp1-var*:$\Delta_1\ x\ y \Longrightarrow Tgt_0\ (x \odot_1\ y) = \{\tau_0\ y\}$
⟨*proof*⟩

We lift the axioms to the powerset level.

lemma *comm-S0S1*: $Src_0\ (Src_1\ X) = Src_1\ (Src_0\ X)$
⟨*proof*⟩

lemma *comm-T0T1*: $Tgt_0\ (Tgt_1\ X) = Tgt_1\ (Tgt_0\ X)$
⟨*proof*⟩

lemma *comm-S0T1*: $Src_0\ (Tgt_1\ x) = Tgt_1\ (Src_0\ x)$
⟨*proof*⟩

lemma *comm-T0S1*: $Tgt_0\ (Src_1\ x) = Src_1\ (Tgt_0\ x)$
⟨*proof*⟩

lemma *interchange-lifting*: $(W *_1 X) *_0 (Y *_1 Z) \subseteq (W *_0 Y) *_1 (X *_0 Z)$
⟨*proof*⟩

lemma *Src1-hom*: $Src_1\ (X *_0 Y) \subseteq Src_1\ X *_0 Src_1\ Y$
⟨*proof*⟩

lemma *Tgt1-hom*: $Tgt_1\ (X *_0 Y) \subseteq Tgt_1\ X *_0 Tgt_1\ Y$
⟨*proof*⟩

lemma *Src0-hom*: $Src_0\ (X *_1 Y) \subseteq Src_0\ X *_1 Src_0\ Y$
⟨*proof*⟩

lemma *Tgt0-hom*: $Tgt_0\ (X *_1 Y) \subseteq Tgt_0\ X *_1 Tgt_0\ Y$
⟨*proof*⟩

lemma *S1S0* [*simp*]: $Src_1\ (Src_0\ X) = Src_0\ X$
⟨*proof*⟩

lemma *S1T0* [*simp*]: $Src_1\ (Tgt_0\ X) = Tgt_0\ X$
⟨*proof*⟩

lemma *T1S0* [*simp*]: $Tgt_1\ (Src_0\ X) = Src_0\ X$
⟨*proof*⟩

lemma *T1T0* [*simp*]: $Tgt_1\ (Tgt_0\ X) = Tgt_0\ X$
⟨*proof*⟩

lemma (**in** *two-st-multimagma*)
s1fix $*_0$ *s1fix* ⊆ *s1fix*

⟨*proof*⟩

lemma *id1-comp0-eq*: *s1fix* ⊆ *s1fix* $*_0$ *s1fix*
⟨*proof*⟩

lemma (**in** *two-st-multimagma*) *id01*:
s0fix ⊆ *s1fix*
⟨*proof*⟩

end

context *two-st-multimagma-strong*
begin

lemma *Src1-hom-strong*: Src_1 ($X *_0 Y$) = Src_1 $X *_0$ Src_1 Y
⟨*proof*⟩

lemma *Tgt1-hom-strong*: Tgt_1 ($X *_0 Y$) = Tgt_1 $X *_0$ Tgt_1 Y
⟨*proof*⟩

lemma *id1-comp0*: *s1fix* $*_0$ *s1fix* ⊆ *s1fix*
⟨*proof*⟩

lemma *id1-comp0-eq* [*simp*]: *s1fix* $*_0$ *s1fix* = *s1fix*
⟨*proof*⟩

end

2.3 2-Catoids and single-set 2-categories

class *two-catoid* = *two-st-multimagma* + *catoid0* + *catoid1*

lemma (**in** *two-catoid*) $\Delta_0\ x\ y \Longrightarrow Src_1\ (x \odot_0 y) = \{\sigma_1\ x\}$

⟨*proof*⟩

lemma (**in** *two-catoid*) $\Delta_0\ x\ y \Longrightarrow Tgt_1\ (x \odot_0 y) = \{\tau_1\ x\}$

⟨*proof*⟩

class *two-catoid-strong* = *two-st-multimagma-strong* + *catoid0* + *catoid1*

class *local-two-catoid* = *two-st-multimagma* + *local-catoid0* + *local-catoid1*

begin

local 2-catoids need not be strong

lemma $Src_1\ (x \odot_0\ y) = \sigma_1\ x \odot_0 \sigma_1\ y$

⟨*proof*⟩

lemma $Tgt_1\ (x \odot_0\ y) = \tau_1\ x \odot_0 \tau_1\ y$

⟨*proof*⟩

lemma $Src_1\ (x \odot_0\ y) = \sigma_1\ x \odot_0 \sigma_1\ y \vee Tgt_1\ (x \odot_0\ y) = \tau_1\ x \odot_0 \tau_1\ y$

⟨*proof*⟩

end

class *functional-two-catoid* = *two-st-multimagma* + *functional-catoid0* + *functional-catoid1*

begin

lemma $Src_1\ (x \odot_0\ y) = \sigma_1\ x \odot_0 \sigma_1\ y$

⟨*proof*⟩

lemma $Tgt_1\ (x \odot_0\ y) = \tau_1\ x \odot_0 \tau_1\ y$

⟨*proof*⟩

lemma $Src_1\ (x \odot_0\ y) = \sigma_1\ x \odot_0 \sigma_1\ y \vee Tgt_1\ (x \odot_0\ y) = \tau_1\ x \odot_0 \tau_1\ y$

⟨*proof*⟩

end

class *local-two-catoid-strong* = *two-st-multimagma-strong* + *local-catoid0* + *local-catoid1*

class *two-category* = *two-st-multimagma* + *single-set-category0* + *single-set-category1*

begin

lemma *s1-hom-strong* [*simp*]: $Src_1\ (x \odot_0\ y) = \sigma_1\ x \odot_0 \sigma_1\ y$
⟨*proof*⟩

lemma *s1-hom-strong-delta*: $\Delta_0\ x\ y = \Delta_0\ (\sigma_1\ x)\ (\sigma_1\ y)$
⟨*proof*⟩

lemma *t1-hom-strong* [*simp*]: $Tgt_1\ (x \odot_0\ y) = \tau_1\ x \odot_0 \tau_1\ y$
⟨*proof*⟩

lemma *t1-hom-strong-delta*: $\Delta_0\ x\ y = \Delta_0\ (\tau_1\ x)\ (\tau_1\ y)$
⟨*proof*⟩

lemma *conv0-sgl*: $a \in x \odot_0 y \Longrightarrow \{a\} = x \odot_0 y$
⟨*proof*⟩

lemma *conv1-sgl*: $a \in \{x\} *_1 \{y\} \Longrightarrow \{a\} = \{x\} *_1 \{y\}$
⟨*proof*⟩

Next we derive some simple globular properties.

lemma *strong-interchange-St1*:
assumes $a \in (w \odot_0 x) *_1 (y \odot_0 z)$
shows $Tgt_1\ (w \odot_0 x) = Src_1\ (y \odot_0 z)$
⟨*proof*⟩

lemma *strong-interchange-ll0*:
assumes $a \in (w \odot_0 x) *_1 (y \odot_0 z)$
shows $\sigma_0\ w = \sigma_0\ y$
⟨*proof*⟩

There is no strong interchange law, and the homomorphism laws for zero sources and targets stay weak, too.

lemma $(w \odot_1 y) *_0 (x \odot_1 z) = (w \odot_0 x) *_1 (y \odot_0 z)$

⟨*proof*⟩

lemma $R_0\ (x \odot_1 y) = r_0\ x \odot_1 r_0\ y$

⟨*proof*⟩

lemma $L_0\ (x \odot_1 y) = l_0\ x \odot_1 l_0\ y$

⟨*proof*⟩

lemma $(W *_0 Y) *_1 (X *_0 Z) = (W *_1 X) *_0 (Y *_1 Z)$

⟨*proof*⟩

lemma $\Delta_0\ x\ y \Longrightarrow Src_1\ (x \odot_0 y) = \{\sigma_1\ x\}$

⟨*proof*⟩

lemma $\Delta_0\ x\ y \Longrightarrow Tgt_1\ (x \odot_0 y) = \{\tau_1\ x\}$

⟨*proof*⟩

end

2.4 Reduced axiomatisations

class *two-st-multimagma-red* = *st-multimagma0* + *st-multimagma1* +
assumes *interchange*: $(w \odot_1 x) *_0 (y \odot_1 z) \subseteq (w \odot_0 y) *_1 (x \odot_0 z)$
assumes *src1-hom*: $Src_1\ (x \odot_0 y) = \sigma_1\ x \odot_0 \sigma_1\ y$
and *tgt1-hom*: $Tgt_1\ (x \odot_0 y) = \tau_1\ x \odot_0 \tau_1\ y$
and *src0-weak-hom*: $Src_0\ (x \odot_1 y) \subseteq \sigma_0\ x \odot_1 \sigma_0\ y$
and *tgt0-weak-hom*: $Tgt_0\ (x \odot_1 y) \subseteq \sigma_0\ x \odot_1 \sigma_0\ y$

begin

lemma *s0t1s0* [*simp*]: $\sigma_0\ (\tau_1\ (\sigma_0\ x)) = \sigma_0\ x$
⟨*proof*⟩

lemma *t0s1s0* [*simp*]: $\tau_0\ (\sigma_1\ (\sigma_0\ x)) = \sigma_0\ x$
⟨*proof*⟩

lemma *s1s0* [*simp*]: $\sigma_1\ (\sigma_0\ x) = \sigma_0\ x$
⟨*proof*⟩

lemma *s1t0* [*simp*]: $\sigma_1\ (\tau_0\ x) = \tau_0\ x$
⟨*proof*⟩

lemma *t1s0* [*simp*]: $\tau_1\ (\sigma_0\ x) = \sigma_0\ x$
⟨*proof*⟩

lemma *t1t0* [*simp*]: $\tau_1\ (\tau_0\ x) = \tau_0\ x$
⟨*proof*⟩

lemma *comm-s0s1*: $\sigma_0\ (\sigma_1\ x) = \sigma_1\ (\sigma_0\ x)$
⟨*proof*⟩

lemma *comm-s0t1*: $\sigma_0\ (\tau_1\ x) = \tau_1\ (\sigma_0\ x)$
⟨*proof*⟩

lemma *comm-t0s1*: $\tau_0\ (\sigma_1\ x) = \sigma_1\ (\tau_0\ x)$
⟨*proof*⟩

lemma *comm-t0t1*: $\tau_0\ (\tau_1\ x) = \tau_1\ (\tau_0\ x)$
⟨*proof*⟩

lemma $\sigma_0\ x = \sigma_1\ x$

⟨*proof*⟩

lemma $\sigma_0\ x = \tau_1\ x$

⟨*proof*⟩

lemma $\tau_0\ x = \tau_1\ x$

⟨*proof*⟩

lemma $\sigma_0\ x = \tau_0\ x$

⟨*proof*⟩

lemma $\sigma_1\ x = \tau_1\ x$

⟨*proof*⟩

lemma $x \odot_0 y = x \odot_1 y$

⟨*proof*⟩

lemma $x \odot_0 y = y \odot_0 x$

⟨*proof*⟩

lemma $x \odot_1 y = y \odot_1 x$

⟨*proof*⟩

end

class *two-catoid-red* = *catoid0* + *catoid1* +
assumes *interchange*: $(w \odot_1 x) *_0 (y \odot_1 z) \subseteq (w \odot_0 y) *_1 (x \odot_0 z)$
and *s1-hom*: $Src_1\ (x \odot_0 y) \subseteq \sigma_1\ x \odot_0 \sigma_1\ y$
and *t1-hom*: $Tgt_1\ (x \odot_0 y) \subseteq \tau_1\ x \odot_0 \tau_1\ y$

begin

lemma *s0t1s0* [*simp*]: $\sigma_0\ (\tau_1\ (\sigma_0\ x)) = \sigma_0\ x$
⟨*proof*⟩

lemma *t0s1s0* [*simp*]: $\tau_0\ (\sigma_1\ (\sigma_0\ x)) = \sigma_0\ x$
⟨*proof*⟩

lemma *s1s0* [*simp*]: $\sigma_1\ (\sigma_0\ x) = \sigma_0\ x$
⟨*proof*⟩

lemma *s1t0* [*simp*]: $\sigma_1\ (\tau_0\ x) = \tau_0\ x$
⟨*proof*⟩

lemma *t1s0* [*simp*]: $\tau_1\ (\sigma_0\ x) = \sigma_0\ x$
⟨*proof*⟩

lemma *t1t0* [*simp*]: $\tau_1\ (\tau_0\ x) = \tau_0\ x$
⟨*proof*⟩

lemma *comm-s0s1*: $\sigma_0\ (\sigma_1\ x) = \sigma_1\ (\sigma_0\ x)$
⟨*proof*⟩

lemma *comm-s0t1*: $\sigma_0\ (\tau_1\ x) = \tau_1\ (\sigma_0\ x)$
⟨*proof*⟩

lemma *comm-t0s1*: $\tau_0\ (\sigma_1\ x) = \sigma_1\ (\tau_0\ x)$
⟨*proof*⟩

lemma *comm-t0t1*: $\tau_0\ (\tau_1\ x) = \tau_1\ (\tau_0\ x)$
⟨*proof*⟩

lemma *s0-hom*: $Src_0\ (x \odot_1\ y) \subseteq \sigma_0\ x \odot_1\ \sigma_0\ y$
⟨*proof*⟩

lemma *t0-hom*: $Tgt_0\ (x \odot_1\ y) \subseteq \tau_0\ x \odot_1\ \tau_0\ y$
⟨*proof*⟩

end

class *two-catoid-red-strong* = *catoid0* + *catoid1* +
assumes *interchange*: $(w \odot_1\ x) *_0 (y \odot_1\ z) \subseteq (w \odot_0\ y) *_1 (x \odot_0\ z)$
and *s1-hom-strong*: $Src_1\ (x \odot_0\ y) = \sigma_1\ x \odot_0\ \sigma_1\ y$
and *t1-hom-strong*: $Tgt_1\ (x \odot_0\ y) = \tau_1\ x \odot_0\ \tau_1\ y$

begin

lemma *s0t1s0* [*simp*]: $\sigma_0\ (\tau_1\ (\sigma_0\ x)) = \sigma_0\ x$
⟨*proof*⟩

lemma *t0s1s0* [*simp*]: $\tau_0\ (\sigma_1\ (\sigma_0\ x)) = \sigma_0\ x$
⟨*proof*⟩

lemma *s1s0* [*simp*]: $\sigma_1\ (\sigma_0\ x) = \sigma_0\ x$
⟨*proof*⟩

lemma *s1t0* [*simp*]: $\sigma_1\ (\tau_0\ x) = \tau_0\ x$
⟨*proof*⟩

lemma *t1s0* [*simp*]: $\tau_1\ (\sigma_0\ x) = \sigma_0\ x$
⟨*proof*⟩

lemma *t1t0* [*simp*]: $\tau_1\ (\tau_0\ x) = \tau_0\ x$
⟨*proof*⟩

lemma *comm-s0s1*: $\sigma_0\ (\sigma_1\ x) = \sigma_1\ (\sigma_0\ x)$
⟨*proof*⟩

lemma *comm-s0t1*: $\sigma_0\ (\tau_1\ x) = \tau_1\ (\sigma_0\ x)$
⟨*proof*⟩

lemma *comm-t0s1*: $\tau_0\ (\sigma_1\ x) = \sigma_1\ (\tau_0\ x)$
⟨*proof*⟩

lemma *comm-t0t1*: $\tau_0\ (\tau_1\ x) = \tau_1\ (\tau_0\ x)$
⟨*proof*⟩

lemma *s0-weak-hom*: $Src_0\ (x \odot_1 y) \subseteq \sigma_0\ x \odot_1 \sigma_0\ y$
⟨*proof*⟩

lemma *t0-weak-hom*: $Tgt_0\ (x \odot_1 y) \subseteq \tau_0\ x \odot_1 \tau_0\ y$
⟨*proof*⟩

end

class *two-catoid-red2* = *single-set-category0* + *single-set-category1* +
assumes *comm-s0s1*: $\sigma_0\ (\sigma_1\ x) = \sigma_1\ (\sigma_0\ x)$
and *comm-s0t1*: $\sigma_0\ (\tau_1\ x) = \tau_1\ (\sigma_0\ x)$
and *comm-t0s1*: $\tau_0\ (\sigma_1\ x) = \sigma_1\ (\tau_0\ x)$
and *comm-t0t1*: $\tau_0\ (\tau_1\ x) = \tau_1\ (\tau_0\ x)$
and *s1s0* [*simp*]: $\sigma_1\ (\sigma_0\ x) = \sigma_0\ x$
and *s1t0* [*simp*]: $\sigma_1\ (\tau_0\ x) = \tau_0\ x$
and *t1s0* [*simp*]: $\tau_1\ (\sigma_0\ x) = \sigma_0\ x$
and *t1t0* [*simp*]: $\tau_1\ (\tau_0\ x) = \tau_0\ x$

begin

lemma $(w \odot_1 x) *_0 (y \odot_1 z) \subseteq (w \odot_0 y) *_1 (x \odot_0 z)$

⟨*proof*⟩

lemma $Src_1\ (x \odot_0 y) \subseteq \sigma_1\ x \odot_0 \sigma_1\ y$

⟨*proof*⟩

lemma $Tgt_1\ (x \odot_0 y) \subseteq \tau_1\ x \odot_0 \tau_1\ y$

⟨*proof*⟩

lemma *s0-hom*: $Src_0\ (x \odot_1 y) \subseteq \sigma_0\ x \odot_1 \sigma_0\ y$
⟨*proof*⟩

lemma *t0-hom*: $Tgt_0\ (x \odot_1 y) \subseteq \tau_0\ x \odot_1 \tau_0\ y$
⟨*proof*⟩

end

class *two-catoid-red3* = *catoid0* + *catoid1* +
  assumes *interchange*: $(w \odot_1 x) *_0 (y \odot_1 z) \subseteq (w \odot_0 y) *_1 (x \odot_0 z)$
  and *s1-hom*: $Src_0\ (x \odot_1 y) \subseteq \sigma_0\ x \odot_1 \sigma_0\ y$
  and *t1-hom*: $Tgt_0\ (x \odot_1 y) \subseteq \tau_0\ x \odot_1 \tau_0\ y$

lemma (**in** *two-catoid-red3*)
  $Src_1\ (x \odot_0 y) \subseteq \sigma_1\ x \odot_0 \sigma_1\ y$

  ⟨*proof*⟩

lemma (**in** *two-catoid-red3*)
  $Tgt_1\ (x \odot_0 y) \subseteq \tau_1\ x \odot_0 \tau_1\ y$

  ⟨*proof*⟩

end

3 2-Kleene algebras

theory *Two-Kleene-Algebra*
  imports *Quantales-Converse.Modal-Kleene-Algebra-Var*

begin

Here we develop (globular) 2-semigroups and (globular) 2-Kleene algebras. These should eventually be extended to n-structures and omega-structures.

3.1 Copies for 0-structures

class *comp0-op* =
  fixes *comp0* :: $'a \Rightarrow 'a \Rightarrow 'a$ (**infixl** $\cdot_0$ *70*)

class *id0-op* =
  fixes *id0* :: $'a$ (1_0)

class *star0-op* =
  fixes *star0* :: $'a \Rightarrow 'a$

class *dom0-op* =
  fixes dom_0 :: $'a \Rightarrow 'a$

class *cod0-op* =
  fixes cod_0 :: $'a \Rightarrow 'a$

class *monoid-mult0* = *comp0-op* + *id0-op* +
  assumes *par-assoc0*: $x \cdot_0 (y \cdot_0 z) = (x \cdot_0 y) \cdot_0 z$
  and *comp0-unl*: $1_0 \cdot_0 x = x$
  and *comp0-unr*: $x \cdot_0 1_0 = x$

class *dioid0* = *monoid-mult0* + *join-semilattice-zero* +
  assumes *distl0*: $x \cdot_0 (y + z) = x \cdot_0 y + x \cdot_0 z$
  and *distr0*: $(x + y) \cdot_0 z = x \cdot_0 z + y \cdot_0 z$
  and *annil0*: $0 \cdot_0 x = 0$
  and *annir0*: $x \cdot_0 0 = 0$

class *kleene-algebra0* = *dioid0* + *star0-op* +
  assumes *star0-unfoldl*: $1_0 + x \cdot_0 \mathit{star0}\ x \leq \mathit{star0}\ x$
  and *star0-unfoldr*: $1_0 + \mathit{star0}\ x \cdot_0 x \leq \mathit{star0}\ x$
  and *star0-inductl*: $z + x \cdot_0 y \leq y \Longrightarrow \mathit{star0}\ x \cdot_0 z \leq y$
  and *star0-inductr*: $z + y \cdot_0 x \leq y \Longrightarrow z \cdot_0 \mathit{star0}\ x \leq y$

class *domain-semiring0* = *dioid0* + *dom0-op* +
  assumes *d0-absorb*: $x \leq dom_0\ x \cdot_0 x$
  and *d0-local*: $dom_0\ (x \cdot_0 dom_0\ y) = dom_0\ (x \cdot_0 y)$
  and *d0-add*: $dom_0\ (x + y) = dom_0\ x + dom_0\ y$
  and *d0-subid*: $dom_0\ x \leq 1_0$
  and *d0-zero*: $dom_0\ 0 = 0$

class *codomain-semiring0* = *dioid0* + *cod0-op* +
  assumes *cod0-absorb*: $x \leq x \cdot_0 cod_0\ x$
  and *cod0-local*: $cod_0\ (cod_0\ x \cdot_0 y) = cod_0\ (x \cdot_0 y)$
  and *cod0-add*: $cod_0\ (x + y) = cod_0\ x + cod_0\ y$
  and *cod0-subid*: $cod_0\ x \leq 1_0$
  and *cod0-zero*: $cod_0\ 0 = 0$

class *modal-semiring0* = *domain-semiring0* + *codomain-semiring0* +
  assumes *dc-compat0*: $dom_0\ (cod_0\ x) = cod_0\ x$
  and *cd-compat0*: $cod_0\ (dom_0\ x) = dom_0\ x$

class *modal-kleene-algebra0* = *modal-semiring0* + *kleene-algebra0*

sublocale *monoid-mult0* ⊆ *mm0*: *monoid-mult* 1_0 $(\cdot_0)$
  ⟨*proof*⟩

sublocale *dioid0* ⊆ *d0*: *dioid-one-zero* - $(\cdot_0)$ 1_0 - - -
  ⟨*proof*⟩

sublocale *dioid0* ⊆ *dd0*: *dioid0* - - - - $\lambda x\ y.\ y \cdot_0 x$ -
  ⟨*proof*⟩

sublocale *kleene-algebra0* ⊆ *k0*: *kleene-algebra* - $(\cdot_0)$ 1_0 - - - *star0*
  ⟨*proof*⟩

sublocale *kleene-algebra0* ⊆ *dk0*: *kleene-algebra0* - - - - $\lambda x\ y.\ y \cdot_0 x$ - -
  ⟨*proof*⟩

sublocale *domain-semiring0* ⊆ *dsr0*: *domain-semiring* - $(\cdot_0)$ 1_0 - dom_0 - -

⟨*proof*⟩

sublocale *codomain-semiring0* ⊆ *csr0*: *range-semiring* - $(\cdot_0)$ *1*$_0$ - *cod*$_0$ - -
⟨*proof*⟩

sublocale *codomain-semiring0* ⊆ *ds0dual*: *domain-semiring0* - - - - $\lambda x\ y.\ y \cdot_0 x$ - *cod*$_0$
⟨*proof*⟩

sublocale *modal-semiring0* ⊆ *msr0*: *dr-modal-semiring* - $(\cdot_0)$ *1*$_0$ - *dom*$_0$ - - *cod*$_0$
⟨*proof*⟩

sublocale *modal-semiring0* ⊆ *msr0dual*: *modal-semiring0* *dom*$_0$ - - - - $\lambda x\ y.\ y \cdot_0 x$ - *cod*$_0$
⟨*proof*⟩

sublocale *modal-kleene-algebra0* ⊆ *mka0*: *dr-modal-kleene-algebra* - $(\cdot_0)$ *1*$_0$ - - - *star0* *dom*$_0$ *cod*$_0$⟨*proof*⟩

sublocale *modal-kleene-algebra0* ⊆ *mka0dual*: *modal-kleene-algebra0* - - - - $\lambda x\ y.\ y \cdot_0 x$ - - *dom*$_0$ *cod*$_0$⟨*proof*⟩

3.2 Copies for 1-structures

class *comp1-op* =
  fixes *comp1* :: $'a \Rightarrow 'a \Rightarrow 'a$ (**infixl** $\cdot_1$ *70*)

class *id1-op* =
  fixes *id1* :: $'a$ (*1*$_1$)

class *star1-op* =
  fixes *star1* :: $'a \Rightarrow 'a$

class *dom1-op* =
  fixes *dom*$_1$ :: $'a \Rightarrow 'a$

class *cod1-op* =
  fixes *cod*$_1$ :: $'a \Rightarrow 'a$

class *monoid-mult1* = *comp1-op* + *id1-op* +
  assumes *par-assoc1*: $x \cdot_1 (y \cdot_1 z) = (x \cdot_1 y) \cdot_1 z$
  and *comp1-unl*: $1_1 \cdot_1 x = x$
  and *comp1-unr*: $x \cdot_1 1_1 = x$

class *dioid1* = *monoid-mult1* + *join-semilattice-zero* +
  assumes *distl1*: $x \cdot_1 (y + z) = x \cdot_1 y + x \cdot_1 z$
  and *distr1*: $(x + y) \cdot_1 z = x \cdot_1 z + y \cdot_1 z$
  and *annil1*: $0 \cdot_1 x = 0$
  and *annir1*: $x \cdot_1 0 = 0$

class *kleene-algebra1* = *dioid1* + *star1-op* +
  assumes *star1-unfoldl*: $1_1 + x \cdot_1 \mathit{star1}\ x \leq \mathit{star1}\ x$
  and *star1-unfoldr*: $1_1 + \mathit{star1}\ x \cdot_1 x \leq \mathit{star1}\ x$
  and *star1-inductl*: $z + x \cdot_1 y \leq y \Longrightarrow \mathit{star1}\ x \cdot_1 z \leq y$
  and *star1-inductr*: $z + y \cdot_1 x \leq y \Longrightarrow z \cdot_1 \mathit{star1}\ x \leq y$

class *domain-semiring1* = *dioid1* + *dom1-op* +
  assumes *d1-absorb*: $x \leq dom_1\ x \cdot_1 x$
  and *d1-local*: $dom_1\ (x \cdot_1 dom_1\ y) = dom_1\ (x \cdot_1 y)$
  and *d1-add*: $dom_1\ (x + y) = dom_1\ x + dom_1\ y$
  and *d1-subid*: $dom_1\ x \leq 1_1$
  and *d1-zero*: $dom_1\ 0 = 0$

class *codomain-semiring1* = *dioid1* + *cod1-op* +
  assumes *cod1-absorb*: $x \leq x \cdot_1 cod_1\ x$
  and *cod1-local*: $cod_1\ (cod_1\ x \cdot_1 y) = cod_1\ (x \cdot_1 y)$
  and *cod1-add*: $cod_1\ (x + y) = cod_1\ x + cod_1\ y$
  and *cod1-subid*: $cod_1\ x \leq 1_1$
  and *cod1-zero*: $cod_1\ 0 = 0$

class *modal-semiring1* = *domain-semiring1* + *codomain-semiring1* +
  assumes *dc-compat1*: $dom_1\ (cod_1\ x) = cod_1\ x$
  and *cd-compat1*: $cod_1\ (dom_1\ x) = dom_1\ x$

class *modal-kleene-algebra1* = *modal-semiring1* + *kleene-algebra1*

sublocale *monoid-mult1* ⊆ *mm1*: *monoid-mult* 1_1 $(\cdot_1)$
  ⟨*proof*⟩

sublocale *dioid1* ⊆*d1*: *dioid-one-zero* - $(\cdot_1)$ 1_1 - - -
  ⟨*proof*⟩

sublocale *dioid1* ⊆ *dd1*: *dioid1* - - - - $\lambda x\ y.\ y \cdot_1 x$ 1_1
  ⟨*proof*⟩

sublocale *kleene-algebra1* ⊆ *k1*: *kleene-algebra* - $(\cdot_1)$ 1_1 - - - *star1*
  ⟨*proof*⟩

sublocale *kleene-algebra1* ⊆ *dk1*: *kleene-algebra1* - - - - $\lambda x\ y.\ y \cdot_1 x$ 1_1 *star1*
  ⟨*proof*⟩

sublocale *domain-semiring1* ⊆ *dsr1*: *domain-semiring* - $(\cdot_1)$ 1_1 - dom_1 - -
  ⟨*proof*⟩

sublocale *codomain-semiring1* ⊆ *csr1*: *range-semiring* - $(\cdot_1)$ 1_1 - cod_1 - -
  ⟨*proof*⟩

sublocale *codomain-semiring1* ⊆ *ds1dual*: *domain-semiring1* - - - - $\lambda x\ y.\ y \cdot_1 x$ -

cod_1
⟨*proof*⟩

sublocale *modal-semiring1* ⊆ *msr1*: *dr-modal-semiring* - $(\cdot_1)$ 1_1 - dom_1 - - cod_1
⟨*proof*⟩

sublocale *modal-semiring1* ⊆ *msr1dual*: *modal-semiring1* dom_1 - - - - $\lambda x\ y.\ y \cdot_1 x$ - cod_1
⟨*proof*⟩

sublocale *modal-kleene-algebra1* ⊆ *mka1*: *dr-modal-kleene-algebra* - $(\cdot_1)$ 1_1 - - - *star1* dom_1 cod_1⟨*proof*⟩

sublocale *modal-kleene-algebra1* ⊆ *mka1dual*: *modal-kleene-algebra1* - - - - $\lambda x\ y.\ y \cdot_1 x$ - - dom_1 cod_1⟨*proof*⟩

3.3 Globular 2-semirings

class *two-semiring* = *modal-semiring0* + *modal-semiring1* +
assumes *interchange*: $(w \cdot_1 x) \cdot_0 (y \cdot_1 z) \leq (w \cdot_0 y) \cdot_1 (x \cdot_0 z)$
and *d1-hom*: $dom_1\ (x \cdot_0 y) \leq dom_1\ x \cdot_0 dom_1\ y$
and *c1-hom*: $cod_1\ (x \cdot_0 y) \leq cod_1\ x \cdot_0 cod_1\ y$
and *d0-hom*: $dom_0\ (x \cdot_1 y) \leq dom_0\ x \cdot_1 dom_0\ y$
and *c0-hom*: $cod_0\ (x \cdot_1 y) \leq cod_0\ x \cdot_1 cod_0\ y$
and *d1d0* [*simp*]: $dom_1\ (dom_0\ x) = dom_0\ x$

class *strong-two-semiring* = *two-semiring* +
assumes *d1-strong-hom* [*simp*]: $dom_1\ (x \cdot_0 y) = dom_1\ x \cdot_0 dom_1\ y$
and *c1-strong-hom*: $cod_1\ (x \cdot_0 y) = cod_1\ x \cdot_0 cod_1\ y$

sublocale *two-semiring* ⊆ *tgsdual*: *two-semiring* dom_0 - - - - $\lambda x\ y.\ y \cdot_0 x$ - cod_0 dom_1 $\lambda x\ y.\ y \cdot_1 x$ - cod_1
⟨*proof*⟩

sublocale *strong-two-semiring* ⊆ *stgsdual*: *strong-two-semiring* dom_0 - - - - $\lambda x\ y.\ y \cdot_0 x$ - cod_0 dom_1 $\lambda x\ y.\ y \cdot_1 x$ - cod_1
⟨*proof*⟩

context *two-semiring*
begin

lemma *c1d0* [*simp*]: $cod_1\ (dom_0\ x) = dom_0\ x$
⟨*proof*⟩

lemma *d1c0* [*simp*]: $dom_1\ (cod_0\ x) = cod_0\ x$
⟨*proof*⟩

lemma *c1c0* [*simp*]: $cod_1\ (cod_0\ x) = cod_0\ x$
⟨*proof*⟩

lemma $1_1 \cdot_0 1_1 \leq 1_1$

⟨*proof*⟩

lemma *id1-comp0-var*: $1_1 \leq 1_1 \cdot_0 1_1$
⟨*proof*⟩

lemma $1_1 \cdot_0 1_1 = 1_1$

⟨*proof*⟩

lemma *id0-le-id1*: $1_0 \leq 1_1$
⟨*proof*⟩

lemma *id0-comp1-eq* [*simp*]: $1_0 \cdot_1 1_0 = 1_0$
⟨*proof*⟩

lemma *d1-id0* [*simp*]: $dom_1\ 1_0 = 1_0$
⟨*proof*⟩

lemma *d0-id1* [*simp*]: $dom_0\ 1_1 = 1_0$
⟨*proof*⟩

lemma *c0-id1*: $cod_0\ 1_1 = 1_0$
⟨*proof*⟩

lemma *c0-id0*: $cod_1\ 1_0 = 1_0$
⟨*proof*⟩

lemma *comm-d0d1*: $dom_0\ (dom_1\ x) = dom_1\ (dom_0\ x)$
⟨*proof*⟩

lemma *comm-d0c1*: $dom_0\ (cod_1\ x) = cod_1\ (dom_0\ x)$
⟨*proof*⟩

lemma *comm-c0c1*: $cod_0\ (cod_1\ x) = cod_1\ (cod_0\ x)$
⟨*proof*⟩

lemma *comm-c0d1*: $cod_0\ (dom_1\ x) = dom_1\ (cod_0\ x)$
⟨*proof*⟩

We prove further lemmas that are not related to the globular structure.

lemma *d0-comp1-idem* [*simp*]: $dom_0\ x \cdot_1 dom_0\ x = dom_0\ x$
⟨*proof*⟩

lemma *cod0-comp1-idem*: $cod_0\ x \cdot_1 cod_0\ x = cod_0\ x$
⟨*proof*⟩

lemma *dom01-loc* [*simp*]: dom_0 $(x \cdot_1 dom_1\ y) = dom_0\ (x \cdot_1 y)$
⟨*proof*⟩

lemma *cod01-locl*: $cod_0\ (cod_1\ x \cdot_1 y) = cod_0\ (x \cdot_1 y)$
⟨*proof*⟩

lemma *dom01-exp* [*simp*]: $dom_0\ (cod_1\ x \cdot_1 y) = dom_0\ (x \cdot_1 y)$
⟨*proof*⟩

lemma *cod01-exo*: $cod_0\ (x \cdot_1 dom_1\ y) = cod_0\ (x \cdot_1 y)$
⟨*proof*⟩

lemma *dom01-loc-var* [*simp*]: $dom_0\ (x \cdot_0 dom_1\ y) = dom_0\ (x \cdot_0 y)$
⟨*proof*⟩

lemma *cod01-loc-var*: $cod_0\ (cod_1\ x \cdot_0 y) = cod_0\ (x \cdot_0 y)$
⟨*proof*⟩

lemma *dom0cod1-exp*: $dom_0\ (x \cdot_0 y) \leq dom_0\ (cod_1\ x \cdot_0 y)$
⟨*proof*⟩

lemma *cod0dom1-exp*: $cod_0\ (x \cdot_0 y) \leq cod_0\ (x \cdot_0 dom_1\ y)$
⟨*proof*⟩

lemma (**in** *two-semiring*) *d0-comp1*: $dom_0\ x \cdot_0 (y \cdot_1 z) \leq (dom_0\ x \cdot_0 y) \cdot_1 (dom_0\ x \cdot_0 z)$
⟨*proof*⟩

lemma *d1-comp1*: $dom_1\ x \cdot_0 (y \cdot_1 z) \leq (dom_1\ x \cdot_0 y) \cdot_1 (dom_1\ x \cdot_0 z)$
⟨*proof*⟩

lemma *d01-export*: $dom_0\ (dom_1\ x \cdot_1 y) \leq dom_0\ x \cdot_1 dom_0\ y$
⟨*proof*⟩

lemma *cod01-export*: $cod_0\ (x \cdot_1 cod_1\ y) \leq cod_0\ x \cdot_1 cod_0\ y$
⟨*proof*⟩

lemma *d10-export* [*simp*]: $dom_1\ (dom_0\ x \cdot_1 y) = dom_0\ x \cdot_1 dom_1\ y$
⟨*proof*⟩

lemma *cod10-export*: $cod_1\ (x \cdot_1 cod_0\ y) = cod_1\ x \cdot_1 cod_0\ y$
⟨*proof*⟩

lemma *d0-comp10*: $dom_0\ x \cdot_1 dom_0\ y = dom_0\ x \cdot_0 dom_0\ y$
⟨*proof*⟩

lemma *dom-exchange-strong*: $(dom_0\ w \cdot_1 dom_0\ x) \cdot_0 (dom_0\ y \cdot_1 dom_0\ z) = (dom_0\ w \cdot_0 dom_0\ y) \cdot_1 (dom_0\ x \cdot_0 dom_0\ z)$
⟨*proof*⟩

end

context *strong-two-semiring*
begin

lemma *id1-comp0*: $1_1 \cdot_0 1_1 \leq 1_1$
⟨*proof*⟩

lemma *id1-comp0-eq* [*simp*]: $1_1 \cdot_0 1_1 = 1_1$
⟨*proof*⟩

lemma $1_0 = 1_1$

⟨*proof*⟩

lemma *dom0cod1-exp*: $dom_0\ (x \cdot_0 y) = dom_0\ (cod_1\ x \cdot_0 y)$
⟨*proof*⟩

lemma *cod0dom1-exp*: $cod_0\ (x \cdot_0 dom_1\ y) = cod_0\ (x \cdot_0 y)$
⟨*proof*⟩

end

The following laws are diamond laws. It remains to define diamonds for them.

context *two-semiring*
begin

lemma *fdia0fdia1-prop*: $dom_0\ (y \cdot_0 dom_1\ (x \cdot_1 z)) = dom_0\ (y \cdot_0 (x \cdot_1 z))$
⟨*proof*⟩

lemma *bdia0fdia1-prop* [*simp*]: $cod_0\ (dom_1\ (x \cdot_1 z) \cdot_0 y) = cod_0\ ((x \cdot_1 z) \cdot_0 y)$
⟨*proof*⟩

lemma *fdia0bdia1-prop*: $dom_0\ (y \cdot_0 cod_1\ (x \cdot_1 z)) = dom_0\ (y \cdot_0 (x \cdot_1 z))$
⟨*proof*⟩

lemma *bdia0bdia1-prop*: $cod_0\ (cod_1\ (x \cdot_1 z) \cdot_0 y) = cod_0\ ((x \cdot_1 z) \cdot_0 y)$
⟨*proof*⟩

lemma *fdia0fdia1-prop2*: $dom_0\ (y \cdot_0 dom_1\ (x \cdot_1 z)) \leq dom_0\ (y \cdot_0 (dom_0\ x \cdot_1 dom_0\ z))$
⟨*proof*⟩

lemma *fdia00-prop2*: $dom_0\ (y \cdot_0 dom_0\ (x \cdot_1 z)) \leq dom_0\ (y \cdot_0 (dom_0\ x \cdot_1 dom_0\ z))$
⟨*proof*⟩

lemma *bdia0dom1-prop2*: cod_0 $(dom_1$ $(x \cdot_1 z) \cdot_0 y) \leq cod_0$ $((cod_0$ $x \cdot_1 cod_0$ $z) \cdot_0$ $y)$
⟨*proof*⟩

lemma *bdia0dom0-prop2*: cod_0 $(dom_0$ $(x \cdot_1 z) \cdot_0 y) \leq cod_0$ $((dom_0$ $x \cdot_1 dom_0$ $z)$ $\cdot_0$ $y)$
⟨*proof*⟩

lemma *fdia0bdia1-prop-2*: dom_0 $(y \cdot_0 cod_1$ $(z \cdot_1 x)) \leq dom_0$ $(y \cdot_0 (dom_0$ $x \cdot_1 dom_0$ $z))$
⟨*proof*⟩

lemma *fdia0bdiao-prop2*: dom_0 $(y \cdot_0 cod_0$ $(z \cdot_1 x)) \leq dom_0$ $(y \cdot_0 (cod_0$ $z \cdot_1 cod_0$ $x))$
⟨*proof*⟩

lemma *bdia0bdia1-prop2*: cod_0 $(cod_1$ $(z \cdot_1 x) \cdot_0 y) \leq cod_0$ $((cod_0$ $x \cdot_1 cod_0$ $z) \cdot_0 y)$
⟨*proof*⟩

lemma *bdia0bdia0-prop2*: cod_0 $(cod_0$ $(x \cdot_1 z) \cdot_0 y) \leq cod_0$ $((cod_0$ $x \cdot_1 cod_0$ $z) \cdot_0 y)$
⟨*proof*⟩

lemma *fdia1fdia0-prop3* [*simp*]: dom_1 $(x \cdot_1 dom_0$ $(y \cdot_0 z)) \leq dom_1$ $(x \cdot_1 dom_0$ $(dom_1$ $y \cdot_0 z))$
⟨*proof*⟩

lemma *fdia1bdia0-prop3* [*simp*]: dom_1 $(x \cdot_1 cod_0$ $(z \cdot_0 y)) \leq dom_1$ $(x \cdot_1 cod_0$ $(z \cdot_0$ dom_1 $y))$
⟨*proof*⟩

lemma *bdia1fdia0-prop3*: cod_1 $(dom_0$ $(y \cdot_0 z) \cdot_1 x) \leq cod_1$ $(dom_0$ $(cod_1$ $y \cdot_0 z) \cdot_1$ $x)$
⟨*proof*⟩

lemma *bdia1bdia0-prop3*: cod_1 $(cod_0$ $(z \cdot_0 y) \cdot_1 x) \leq cod_1$ $(cod_0$ $(z \cdot_0 cod_1$ $y) \cdot_1 x)$
⟨*proof*⟩

end

context *strong-two-semiring*
begin

lemma *fdia1fdia0-prop3* [*simp*]: dom_1 $(x \cdot_1 dom_0$ $(dom_1$ $y \cdot_0 z)) = dom_1$ $(x \cdot_1$ dom_0 $(y \cdot_0 z))$
⟨*proof*⟩

lemma *fdia1bdia0-prop3* [*simp*]: dom_1 $(x \cdot_1 cod_0$ $(z \cdot_0 dom_1$ $y)) = dom_1$ $(x \cdot_1 cod_0$ $(z \cdot_0 y))$
⟨*proof*⟩

lemma *bdia1fdia0-prop3*: cod_1 $(dom_0$ $(cod_1$ $y \cdot_0 z) \cdot_1 x) = cod_1$ $(dom_0$ $(y \cdot_0 z) \cdot_1$ $x)$
⟨*proof*⟩

lemma *bdia1bdia0-prop3*: cod_1 $(cod_0$ $(z \cdot_0 cod_1$ $y) \cdot_1 x) = cod_1$ $(cod_0$ $(z \cdot_0 y) \cdot_1$ $x)$
⟨*proof*⟩

lemma *fdia0fdia1-prop4*: dom_0 $z \cdot_0 dom_1$ $(x \cdot_1 y) \leq dom_1$ $((dom_0$ $z \cdot_0 x) \cdot_1 (dom_0$ $z \cdot_0 y))$
⟨*proof*⟩

lemma *fdia0bdia1-prop4*: dom_0 $z \cdot_0 cod_1$ $(y \cdot_1 x) \leq cod_1$ $((dom_0$ $z \cdot_0 y) \cdot_1 (dom_0$ $z \cdot_0 x))$
⟨*proof*⟩

lemma *fdia1fdia1-prop4*: dom_1 $(x \cdot_1 y) \cdot_0 dom_0$ $z \leq dom_1$ $((x \cdot_0 dom_0$ $z) \cdot_1 (y \cdot_0$ dom_0 $z))$
⟨*proof*⟩

lemma *bdia1bdia1-prop4*: cod_1 $(y \cdot_1 x) \cdot_0 dom_0$ $z \leq cod_1$ $((y \cdot_0 dom_0$ $z) \cdot_1 (x \cdot_0$ dom_0 $z))$
⟨*proof*⟩

end

3.4 Globular 2-Kleene algebras

class *two-kleene-algebra* = *two-semiring* + *kleene-algebra0* + *kleene-algebra1*

class *strong-two-kleene-algebra* = *strong-two-semiring* + *kleene-algebra0* + *kleene-algebra1*

lemma (**in** *strong-two-kleene-algebra*) *star1* $x \cdot_0$ *star1* $y \leq$ *star0* $(x \cdot_1 y)$

⟨*proof*⟩

lemma (**in** *strong-two-kleene-algebra*) *star1* $x \cdot_0$ *star1* $y \leq$ *star1* $(x \cdot_1 y)$

⟨*proof*⟩

context *two-kleene-algebra*
begin

lemma *interchange-var1*: $(x \cdot_1 x) \cdot_0 (y \cdot_1 y) \cdot_0 (z \cdot_1 z) \leq (x \cdot_0 y \cdot_0 z) \cdot_1 (x \cdot_0 y \cdot_0 z)$
⟨*proof*⟩

lemma *interchange-var2*: $(x \cdot_1 y) \cdot_0 (x \cdot_1 y) \cdot_0 (x \cdot_1 y) \leq (x \cdot_0 x \cdot_0 x) \cdot_1 (y \cdot_0 y \cdot_0 y)$
⟨*proof*⟩

lemma *star0-comp1*: $star0\ (x \cdot_1 y) \leq star0\ x \cdot_1 star0\ y$
⟨*proof*⟩

end

context *strong-two-kleene-algebra*
begin

lemma $star1\ (x \cdot_1 y) \leq star1\ x \cdot_0 star1\ y$

⟨*proof*⟩

lemma $star1\ x \cdot_0 star1\ y \leq star1\ (x \cdot_0 y)$

⟨*proof*⟩

lemma $star1\ (x \cdot_0 y) \leq star1\ x \cdot_0 star1\ y$

⟨*proof*⟩

lemma $star0\ x \cdot_1 star0\ y \leq star0\ (x \cdot_0 y)$

⟨*proof*⟩

lemma $star0\ (x \cdot_0 y) \leq star0\ x \cdot_1 star0\ y$

⟨*proof*⟩

lemma $star0\ x \cdot_1 star0\ y \leq star0\ (x \cdot_1 y)$

⟨*proof*⟩

lemma (**in** *strong-two-kleene-algebra*) $dom_0\ x \cdot_0 star1\ y \leq star1\ (dom_0\ x \cdot_0 y)$
⟨*proof*⟩

end

class *two-quantale-lmcs* = *modal-semiring0* + *modal-semiring1* +
assumes *interchange*: $(w \cdot_1 x) \cdot_0 (y \cdot_1 z) \leq (w \cdot_0 y) \cdot_1 (x \cdot_0 z)$
and *d1-hom*: $dom_1\ (x \cdot_0 y) = dom_1\ x \cdot_0 dom_1\ y$
and *c1-hom*: $cod_1\ (x \cdot_0 y) = cod_1\ x \cdot_0 cod_1\ y$
and *d1d0* [*simp*]: $dom_1\ (dom_0\ x) = dom_0\ x$
and *c1d0* [*simp*]: $cod_1\ (dom_0\ x) = dom_0\ x$
and *d1c0* [*simp*]: $dom_1\ (cod_0\ x) = cod_0\ x$
and *c1c0* [*simp*]: $cod_1\ (cod_0\ x) = cod_0\ x$

and *d0d1* [*simp*]: dom_0 (dom_1 x) = dom_0 x
and *c0d1* [*simp*]: cod_0 (dom_1 x) = dom_0 x
and *d0c1* [*simp*]: dom_0 (cod_1 x) = cod_0 x
and *c0c1* [*simp*]: cod_0 (cod_1 x) = cod_0 x

begin

lemma dom_0 ($x \cdot_1 y$) $\leq$ dom_0 $x \cdot_1$ dom_0 y

⟨*proof*⟩

lemma cod_0 ($x \cdot_1 y$) $\leq$ cod_0 $x \cdot_1$ cod_0 y

⟨*proof*⟩

end

end

4 2-Quantales

theory *Two-Quantale*
imports *Quantales-Converse.Modal-Quantale Two-Kleene-Algebra*

begin

class *quantale0* = *complete-lattice* + *monoid-mult0* +
assumes *Sup-distl0*: $x \cdot_0 \bigsqcup Y = (\bigsqcup y \in Y.\ x \cdot_0 y)$
assumes *Sup-distr0*: $\bigsqcup X \cdot_0 y = (\bigsqcup x \in X.\ x \cdot_0 y)$

sublocale *quantale0* ⊆ *q0q*: *unital-quantale* 1_0 ($\cdot_0$) - - - - - - - -
⟨*proof*⟩

definition (**in** *quantale0*) *qstar0* = *q0q.qstar*

lemma (**in** *quantale0*) *qstar0-unfold*: *qstar0* x = ($\bigsqcup i.$ *mm0.power* x i)
⟨*proof*⟩

sublocale *quantale0* ⊆ *dq0s0*: *dioid0* ($\sqcup$) ($\leq$) ($<$) $\bot$ ($\cdot_0$) 1_0
⟨*proof*⟩

sublocale *quantale0* ⊆ *dq0ka0*: *kleene-algebra0* ($\sqcup$) ($\leq$) ($<$) $\bot$ ($\cdot_0$) 1_0 *qstar0*
⟨*proof*⟩

class *domain-quantale0* = *quantale0* + *dom0-op* +
assumes *dom0-absorb*: $x \leq dom_0\ x \cdot_0 x$
and *dom0-local*: dom_0 ($x \cdot_0 dom_0\ y$) = dom_0 ($x \cdot_0 y$)
and *dom0-add*: dom_0 ($x \sqcup y$) = $dom_0\ x \sqcup dom_0\ y$
and *dom0-subid*: $dom_0\ x \leq 1_0$

and *dom0-zero*: $dom_0 \perp = \perp$

sublocale *domain-quantale0* $\subseteq$ *dq0dq*: *domain-quantale* dom_0 1_0 $(\cdot_0)$ - - - - - - - -
⟨*proof*⟩

sublocale *domain-quantale0* $\subseteq$ *dq0ds0*: *domain-semiring0* $(\sqcup)$ $(\leq)$ $(<)$ $\perp$ $(\cdot_0)$ 1_0 dom_0
⟨*proof*⟩

class *codomain-quantale0* = *quantale0* + *cod0-op* +
assumes *cod0-absorb*: $x \leq x \cdot_0 cod_0\ x$
and *cod0-local*: $cod_0\ (cod_0\ x \cdot_0 y) = cod_0\ (x \cdot_0 y)$
and *cod0-add*: $cod_0\ (x \sqcup y) = cod_0\ x \sqcup cod_0\ y$
and *cod0-subid*: $cod_0\ x \leq 1_0$
and *cod0-zero*: $cod_0 \perp = \perp$

sublocale *codomain-quantale0* $\subseteq$ *cdq0cdq*: *codomain-quantale* 1_0 $(\cdot_0)$ - - - - - - - - cod_0
⟨*proof*⟩

sublocale *codomain-quantale0* $\subseteq$ *cdq0dcs0*: *codomain-semiring0* cod_0 $(\sqcup)$ $(\leq)$ $(<)$ $\perp$ $(\cdot_0)$ 1_0
⟨*proof*⟩

class *modal-quantale0* = *domain-quantale0* + *codomain-quantale0* +
assumes *dc-compat*: $dom_0\ (cod_0\ x) = cod_0\ x$
and *cd-compat*: $cod_0\ (dom_0\ x) = dom_0\ x$

sublocale *modal-quantale0* $\subseteq$ *mq0mq*: *dc-modal-quantale* 1_0 $(\cdot_0)$ - - - - - - - - cod_0 dom_0
⟨*proof*⟩

sublocale *modal-quantale0* $\subseteq$ *mq0mka*: *modal-kleene-algebra0* $(\sqcup)$ $(\leq)$ $(<)$ $\perp$ $(\cdot_0)$ 1_0 *qstar0* cod_0 dom_0
⟨*proof*⟩

sublocale *modal-quantale0* $\subseteq$ *mq0dual*: *modal-quantale0* dom_0 - - - - - - - - $\lambda x\ y.\ y \cdot_0 x$ - cod_0
⟨*proof*⟩

class *quantale1* = *complete-lattice* + *monoid-mult1* +
assumes *Sup-distl1*: $x \cdot_1 \bigsqcup Y = (\bigsqcup y \in Y.\ x \cdot_1 y)$
assumes *Sup-distr1*: $\bigsqcup X \cdot_1 y = (\bigsqcup x \in X.\ x \cdot_1 y)$

sublocale *quantale1* $\subseteq$ *q1q*: *unital-quantale* 1_1 $(\cdot_1)$ - - - - - - - -
⟨*proof*⟩

definition (**in** *quantale1*) *qstar1* = *q1q.qstar*

lemma (**in** *quantale1*) *qstar1-unfold*: *qstar1* $x = (\bigsqcup i.\ mm1.power\ x\ i)$
⟨*proof*⟩

sublocale *quantale1* ⊆ *dq1s1*: *dioid1* $(\sqcup)$ $(\leq)$ $(<)$ $\bot$ $(\cdot_1)$ 1_1
⟨*proof*⟩

sublocale *quantale1* ⊆ *dq0ka1*: *kleene-algebra1* $(\sqcup)$ $(\leq)$ $(<)$ $\bot$ $(\cdot_1)$ 1_1 *qstar1*
⟨*proof*⟩

class *domain-quantale1* = *quantale1* + *dom1-op* +
assumes *dom1-absorb*: $x \leq dom_1\ x \cdot_1 x$
and *dom1-local*: $dom_1\ (x \cdot_1 dom_1\ y) = dom_1\ (x \cdot_1 y)$
and *dom1-add*: $dom_1\ (x \sqcup y) = dom_1\ x \sqcup dom_1\ y$
and *dom1-subid*: $dom_1\ x \leq 1_1$
and *dom1-zero*: $dom_1\ \bot = \bot$

sublocale *domain-quantale1* ⊆ *dq1dq*: *domain-quantale* dom_1 1_1 $(\cdot_1)$ - - - - - - - -
⟨*proof*⟩

sublocale *domain-quantale1* ⊆ *dq1ds1*: *domain-semiring1* $(\sqcup)$ $(\leq)$ $(<)$ $\bot$ $(\cdot_1)$ 1_1 dom_1
⟨*proof*⟩

class *codomain-quantale1* = *quantale1* + *cod1-op* +
assumes *cod1-absorb*: $x \leq x \cdot_1 cod_1\ x$
and *cod1-local*: $cod_1\ (cod_1\ x \cdot_1 y) = cod_1\ (x \cdot_1 y)$
and *cod1-add*: $cod_1\ (x \sqcup y) = cod_1\ x \sqcup cod_1\ y$
and *cod1-subid*: $cod_1\ x \leq 1_1$
and *cod1-zero*: $cod_1\ \bot = \bot$

sublocale *codomain-quantale1* ⊆ *cdq1cdq*: *codomain-quantale* 1_1 $(\cdot_1)$ - - - - - - - - cod_1
⟨*proof*⟩

sublocale *codomain-quantale1* ⊆ *cdq1dcs1*: *codomain-semiring1* cod_1 $(\sqcup)$ $(\leq)$ $(<)$ $\bot$ $(\cdot_1)$ 1_1
⟨*proof*⟩

class *modal-quantale1* = *domain-quantale1* + *codomain-quantale1* +
assumes *dc-compat*: $dom_1\ (cod_1\ x) = cod_1\ x$
and *cd-compat*: $cod_1\ (dom_1\ x) = dom_1\ x$

sublocale *modal-quantale1* ⊆ *mq1mq*: *dc-modal-quantale* 1_1 $(\cdot_1)$ - - - - - - - - cod_1 dom_1
⟨*proof*⟩

sublocale *modal-quantale1* ⊆ *mq1mka*: *modal-kleene-algebra1* $(\sqcup)$ $(\leq)$ $(<)$ $\bot$ $(\cdot_1)$ 1_1 *qstar1* cod_1 dom_1

⟨*proof*⟩

sublocale *modal-quantale1* ⊆ *mq1dual*: *modal-quantale1* dom_1 - - - - - - - - $\lambda x\ y.$ $y \cdot_1 x$ - cod_1
⟨*proof*⟩

class *two-quantale* = *modal-quantale0* + *modal-quantale1* +
assumes *interchange*: $(w \cdot_1 x) \cdot_0 (y \cdot_1 z) \leq (w \cdot_0 y) \cdot_1 (x \cdot_0 z)$
and *d1-hom*: $dom_1\ (x \cdot_0 y) \leq dom_1\ x \cdot_0\ dom_1\ y$
and *c1-hom*: $cod_1\ (x \cdot_0 y) \leq cod_1\ x \cdot_0\ cod_1\ y$
and *d0-weak-hom*: $dom_0\ (x \cdot_1 y) \leq dom_0\ x \cdot_1\ dom_0\ y$
and *c0-weak-hom*: $cod_0\ (x \cdot_1 y) \leq cod_0\ x \cdot_1\ cod_0\ y$
and *d1d0* [*simp*]: $dom_1\ (dom_0\ x) = dom_0\ x$

class *strong-two-quantale* = *two-quantale* +
assumes *d1-strong-hom* [*simp*]: $dom_1\ (x \cdot_0 y) = dom_1\ x \cdot_0\ dom_1\ y$
and *c1-strong-hom* [*simp*]: $cod_1\ (x \cdot_0 y) = cod_1\ x \cdot_0\ cod_1\ y$

sublocale *two-quantale* ⊆ *tgqs*: *two-semiring* cod_0 (⊔) (≤) (<) ⊥ $(\cdot_0)$ 1_0 dom_0 cod_1 $(\cdot_1)$ 1_1 dom_1
⟨*proof*⟩

sublocale *strong-two-quantale* ⊆ *stgqs*: *strong-two-semiring* cod_0 (⊔) (≤) (<) ⊥ $(\cdot_0)$ 1_0 dom_0 cod_1 $(\cdot_1)$ 1_1 dom_1
⟨*proof*⟩

sublocale *two-quantale* ⊆ *tgqs*: *two-kleene-algebra* (⊔) (≤) (<) ⊥ $(\cdot_0)$ 1_0 *qstar0* $(\cdot_1)$ 1_1 *qstar1* cod_0 dom_0 cod_1 dom_1 ⟨*proof*⟩

sublocale *strong-two-quantale* ⊆ *tgqs*: *strong-two-kleene-algebra* (⊔) (≤) (<) ⊥ $(\cdot_0)$ 1_0 *qstar0* $(\cdot_1)$ 1_1 *qstar1* cod_0 dom_0 cod_1 dom_1 ⟨*proof*⟩

lemma (**in** *strong-two-quantale*) *id0-le-id1*: $1_0 = 1_1$

⟨*proof*⟩

context *two-quantale*
begin

lemma *qstar0-aux*: *mm0.power* $(x \cdot_1 y)$ $i \leq$ *mm0.power* $x\ i \cdot_1$ *mm0.power* $y\ i$
⟨*proof*⟩

lemma *qstar0-oplax*: *qstar0* $(x \cdot_1 y) \leq$ *qstar0* $x \cdot_1$ *qstar0* y
⟨*proof*⟩

lemma *qstar1-distl0*: $x \cdot_0$ (*qstar1* y) = ($\bigsqcup i.\ x \cdot_0$ *mm1.power* $y\ i$)
⟨*proof*⟩

lemma *qstar1-distr0*: (*qstar1* x) $\cdot_0\ y$ = ($\bigsqcup i.$ *mm1.power* $x\ i \cdot_0 y$)

⟨*proof*⟩

lemma *qstar0-distl1*: $x \cdot_1 (qstar0\ y) = (\bigsqcup i.\ x \cdot_1 mm0.power\ y\ i)$
⟨*proof*⟩

lemma *qstar0-distr1*: $(qstar0\ x) \cdot_1 y = (\bigsqcup i.\ mm0.power\ x\ i \cdot_1 y)$
⟨*proof*⟩

lemma *star1-laxl-aux-var*: $dom_0\ x \cdot_0 mm1.power\ y\ i \leq mm1.power\ (dom_0\ x \cdot_0 y)\ i$
⟨*proof*⟩

lemma *star1-laxl-var*: $dom_0\ x \cdot_0 qstar1\ y \leq qstar1\ (dom_0\ x \cdot_0 y)$
⟨*proof*⟩

lemma *star1-laxr-aux-var*: $mm1.power\ x\ i \cdot_0 cod_0\ y \leq mm1.power\ (x \cdot_0 cod_0\ y)\ i$
⟨*proof*⟩

lemma *qstar1-laxr-var*: $qstar1\ x \cdot_0 cod_0\ y \leq qstar1\ (x \cdot_0 cod_0\ y)$
⟨*proof*⟩

lemma *qstar-1-power*: $qstar1\ x \cdot_0 qstar1\ y = (\bigsqcup i\ j.\ mm1.power\ x\ i \cdot_0 mm1.power\ y\ j)$
⟨*proof*⟩

end

context *strong-two-quantale*
begin

lemma *star1-laxl-aux*: $dom_1\ x \cdot_0 mm1.power\ y\ i \leq mm1.power\ (dom_1\ x \cdot_0 y)\ i$
⟨*proof*⟩

lemma *star1-laxl*: $dom_1\ x \cdot_0 qstar1\ y \leq qstar1\ (dom_1\ x \cdot_0 y)$
⟨*proof*⟩

lemma *star1-laxr-aux*: $mm1.power\ x\ i \cdot_0 cod_1\ y \leq mm1.power\ (x \cdot_0 cod_1\ y)\ i$
⟨*proof*⟩

lemma *qstar1-laxr*: $qstar1\ x \cdot_0 cod_1\ y \leq qstar1\ (x \cdot_0 cod_1\ y)$
⟨*proof*⟩

lemma *qstar1-aux*: $mm1.power\ x\ i \cdot_0 mm1.power\ y\ i \leq mm1.power\ (x \cdot_0 y)\ i$
⟨*proof*⟩

lemma $qstar1\ x \cdot_0 qstar1\ y \leq qstar0\ (x \cdot_1 y)$

⟨*proof*⟩

lemma *qstar1* $x \cdot_0$ *qstar1* $y \leq$ *qstar1* $(x \cdot_1 y)$

⟨*proof*⟩

lemma *qstar1* $(x \cdot_1 y) \leq$ *qstar1* $x \cdot_0$ *qstar1* y

⟨*proof*⟩

lemma *qstar1* $x \cdot_0$ *qstar1* $y \leq$ *qstar1* $(x \cdot_0 y)$

⟨*proof*⟩

lemma *qstar1* $(x \cdot_0 y) \leq$ *qstar1* $x \cdot_0$ *qstar1* y

⟨*proof*⟩

lemma *qstar0* $x \cdot_1$ *qstar0* $y \leq$ *qstar0* $(x \cdot_0 y)$

⟨*proof*⟩

end

lemma (**in** *strong-two-kleene-algebra*) *qstar0* $x \cdot_1$ *qstar0* $y \leq$ *qstar0* $(x \cdot_1 y)$

⟨*proof*⟩

lemma (**in** *strong-two-kleene-algebra*) *qstar0* $(x \cdot_1 y) \leq$ *qstar0* $x \cdot_1$ *qstar0* y

⟨*proof*⟩

class *two-quantale-lmcs* = *modal-quantale0* + *modal-quantale1* +
assumes *interchange*: $(w \cdot_1 x) \cdot_0 (y \cdot_1 z) \leq (w \cdot_0 y) \cdot_1 (x \cdot_0 z)$
and *d1-hom*: $dom_1\ (x \cdot_0 y) = dom_1\ x \cdot_0 dom_1\ y$
and *c1-hom*: $cod_1\ (x \cdot_0 y) = cod_1\ x \cdot_0 cod_1\ y$
and *d1d0* [*simp*]: $dom_1\ (dom_0\ x) = dom_0\ x$
and *c1d0* [*simp*]: $cod_1\ (dom_0\ x) = dom_0\ x$
and *d1c0* [*simp*]: $dom_1\ (cod_0\ x) = cod_0\ x$
and *c1c0* [*simp*]: $cod_1\ (cod_0\ x) = cod_0\ x$
and *d0d1* [*simp*]: $dom_0\ (dom_1\ x) = dom_0\ x$
and *c0d1* [*simp*]: $cod_0\ (dom_1\ x) = dom_0\ x$
and *d0c1* [*simp*]: $dom_0\ (cod_1\ x) = cod_0\ x$
and *c0c1* [*simp*]: $cod_0\ (cod_1\ x) = cod_0\ x$

begin

lemma $dom_0\ (x \cdot_1 y) \leq dom_0\ x \cdot_1 dom_0\ y$

⟨*proof*⟩

lemma $cod_0\ (x \cdot_1\ y) \leq cod_0\ x \cdot_1\ cod_0\ y$

⟨*proof*⟩

end

end

5 Lifting 2-Catoids to powerset 2-quantales

theory *Two-Catoid-Lifting*
imports *Two-Catoid Two-Quantale Catoids.Catoid-Lifting*

begin

instantiation *set* :: (*local-two-catoid*) *two-quantale*

begin

definition dom_0-*set* :: *′a set* ⇒ *′a set* **where**
$dom_0\ X = Src_0\ X$

definition cod_0-*set* :: *′a set* ⇒ *′a set* **where**
$cod_0\ X = Tgt_0\ X$

definition *comp0-set* :: *′a set* ⇒ *′a set* ⇒ *′a set* **where**
$X \cdot_0\ Y = X *_0\ Y$

definition *id0-set* :: *′a set*
where $1_0 = s0fix$

definition dom_1-*set* :: *′a set* ⇒ *′a set* **where**
$dom_1\ X = Src_1\ X$

definition cod_1-*set* :: *′a set* ⇒ *′a set* **where**
$cod_1\ X = Tgt_1\ X$

definition *comp1-set* :: *′a set* ⇒ *′a set* ⇒ *′a set* **where**
$X \cdot_1\ Y = X *_1\ Y$

definition *id1-set* :: *′a set* **where**
$1_1 = t1fix$

instance
⟨*proof*⟩

end

end

6 2-Catoids with (too) strong homomorphisms

theory *Two-Catoid-Collapse*
imports *Two-Catoid*

begin

Here we present variants of 2-categories where the axioms are too strong. There is an Eckmann-Hilton style collapse of the two structures.

6.1 2-st-Multimagmas with strong homomorphism laws

class *two-st-multimagma-collapse* = *st-multimagma0* + *st-multimagma1* +
assumes *comm-s0s1*: $\sigma_0\ (\sigma_1\ x) = \sigma_1\ (\sigma_0\ x)$
and *comm-t0t1*: $\tau_0\ (\tau_1\ x) = \tau_1\ (\tau_0\ x)$
and *comm-s0t1*: $\sigma_0\ (\tau_1\ x) = \tau_1\ (\sigma_0\ x)$
and *commtr0s1*: $\tau_0\ (\sigma_1\ x) = \sigma_1\ (\tau_0\ x)$
assumes *interchange*: $(w \odot_1 x) *_0 (y \odot_1 z) \subseteq (w \odot_0 y) *_1 (x \odot_0 z)$
and *t0-hom*: $Tgt_0\ (x \odot_1 y) = \tau_0\ x \odot_1 \tau_0\ y$
and *t1-hom*: $Tgt_1\ (x \odot_0 y) = \tau_1\ x \odot_0 \tau_1\ y$
and *s0-hom*: $Src_0\ (x \odot_1 y) = \sigma_0\ x \odot_1 \sigma_0\ y$
and *s1-hom*: $Src_1\ (x \odot_0 y) = \sigma_1\ x \odot_0 \sigma_1\ y$
and *s1-s0* [*simp*]: $\sigma_1\ (\sigma_0\ x) = \sigma_0\ x$
and *t1-s0* [*simp*]: $\tau_1\ (\sigma_0\ x) = \sigma_0\ x$
and *s1-t0* [*simp*]: $\sigma_1\ (\tau_0\ x) = \tau_0\ x$
and *t1-t0* [*simp*]: $\tau_1\ (\tau_0\ x) = \tau_0\ x$

begin

The source and target structure collapses.

lemma *s0s1*: $\sigma_0\ x = \sigma_1\ x$
⟨*proof*⟩

lemma *t0t1*: $\tau_0\ x = \tau_1\ x$
⟨*proof*⟩

lemma *s0t0*: $\sigma_0\ x = \tau_0\ x$
⟨*proof*⟩

lemma $\sigma_0\ x = x$

⟨*proof*⟩

lemma *s1t1*: $\sigma_1\ x = \tau_1\ x$
⟨*proof*⟩

lemma $x \in y \odot_0 z \Longrightarrow x' \in y \odot_0 z \Longrightarrow x = x'$

⟨*proof*⟩

lemma $x \in y \odot_1 z \Longrightarrow x' \in y \odot_1 z \Longrightarrow x = x'$

⟨*proof*⟩

The two compositions are still different—but see below for 2-catoids.

end

6.2 2-Catoids with (too) strong homomorphisms

class *two-catoid-collapse* = *two-st-multimagma-collapse* + *catoid0* + *catoid1*

begin

The two compositions are still different, and neither of them commutes.

lemma $x \odot_0 y = x \odot_1 y$

⟨*proof*⟩

lemma $x \odot_0 y = y \odot_0 x$

⟨*proof*⟩

lemma $x \odot_1 y = y \odot_1 x$

⟨*proof*⟩

end

6.3 Single-set 2-categories with (too) strong homomorphisms

class *two-category-collapse* = *two-catoid-collapse* + *single-set-category0* + *single-set-category1*

begin

lemma *comp-collapse*: $x \odot_0 y = x \odot_1 y$
⟨*proof*⟩

lemma *comp0-comm*: $x \odot_0 y = y \odot_0 x$
⟨*proof*⟩

lemma *comp1-comm*: $x \odot_1 y = y \odot_1 x$
⟨*proof*⟩

lemma $\sigma_0\ x = x$

⟨*proof*⟩

lemma $\sigma_0\ x = \sigma_0\ y$

⟨*proof*⟩

lemma $x \odot_0 y \neq \{\}$

⟨*proof*⟩

lemma $x \odot_1 y \neq \{\}$

⟨*proof*⟩

end

6.4 2-lr-Multimagmas with strong interchange law

class *two-lr-multimagma-bad* = *st-multimagma0* + *st-multimagma1* +
assumes *comm-s0s1*: $\sigma_0\ (\sigma_1\ x) = \sigma_1\ (\sigma_0\ x)$
and *comm-t0t1*: $\tau_0\ (\tau_1\ x) = \tau_1\ (\tau_0\ x)$
and *comm-s0t1*: $\sigma_0\ (\tau_1\ x) = \tau_1\ (\sigma_0\ x)$
and *comm-t0s1*: $\tau_0\ (\sigma_1\ x) = \sigma_1\ (\tau_0\ x)$
assumes *interchange*: $(w \odot_1 x) *_0 (y \odot_1 z) = (w \odot_0 y) *_1 (x \odot_0 z)$
and *t0-hom*: $Tgt_0\ (x \odot_1 y) = \tau_0\ x \odot_1 \tau_0\ y$
and *t1-hom*: $Tgt_1\ (x \odot_0 y) = \tau_1\ x \odot_0 \tau_1\ y$
and *s0-hom*: $Src_0\ (x \odot_1 y) \subseteq \sigma_0\ x \odot_1 \sigma_0\ y$
and *s1-hom*: $Src_1\ (x \odot_0 y) \subseteq \sigma_1\ x \odot_0 \sigma_1\ y$
and *s1-s0* [*simp*]: $\sigma_1\ (\sigma_0\ x) = \sigma_0\ x$
and *t1-s0* [*simp*]: $\tau_1\ (\sigma_0\ x) = \sigma_0\ x$
and *s1-t0* [*simp*]: $\sigma_1\ (\tau_0\ x) = \tau_0\ x$
and *t1-t0* [*simp*]: $\tau_1\ (\tau_0\ x) = \tau_0\ x$

begin

The source and target structure collapses.

lemma *s0s1*: $\sigma_0\ x = \sigma_1\ x$
⟨*proof*⟩

lemma *t0t1*: $\tau_0\ x = \tau_1\ x$
⟨*proof*⟩

lemma *s0t0*: $\sigma_0\ x = \tau_0\ x$
⟨*proof*⟩

lemma *s1t1*: $\sigma_1\ x = \tau_1\ x$
⟨*proof*⟩

lemma *comp-collapse*: $x \odot_0 y = x \odot_1 y$
⟨*proof*⟩

lemma *comp0-comm*: $x \odot_0 y = y \odot_0 x$
⟨*proof*⟩

lemma *comp1-comm*: $x \odot_1 y = y \odot_1 x$
⟨*proof*⟩

lemma *comp0-assoc*: $\{x\} *_0 (y \odot_0 z) = (x \odot_0 y) *_0 \{z\}$
⟨*proof*⟩

lemma *comp1-assoc*: $\{x\} *_1 (y \odot_1 z) = (x \odot_1 y) *_1 \{z\}$
⟨*proof*⟩

lemma $\sigma_0\ x = x$

⟨*proof*⟩

lemma $\sigma_0\ x = \sigma_0\ y$

⟨*proof*⟩

lemma $x \odot_0 y \neq \{\}$

⟨*proof*⟩

lemma $x \in y \odot_0 z \Longrightarrow x' \in y \odot_0 z \Longrightarrow x = x'$

⟨*proof*⟩

lemma $x \odot_1 y \neq \{\}$

⟨*proof*⟩

lemma $x \in y \odot_1 z \Longrightarrow x' \in y \odot_1 z \Longrightarrow x = x'$

⟨*proof*⟩

end

end

7 ω-Catoids

theory *Omega-Catoid*
imports *Two-Catoid*

begin

7.1 Indexed catoids.

We add an index to the operations of catoids.

class *imultimagma* =
 fixes *imcomp* :: $'a \Rightarrow nat \Rightarrow 'a \Rightarrow 'a\ set$ (-$\odot$ $_{-}$ - [70,70,70] 70)

definition (**in** *imultimagma*) *iconv* :: $'a\ set \Rightarrow nat \Rightarrow 'a\ set \Rightarrow 'a\ set$ (-$\star_{-}$-[70,70,70] 70)
where
 $X \star_i Y = (\bigcup x \in X. \bigcup y \in Y. x \odot_i y)$

class *imultisemigroup* = *imultimagma* +
 assumes *assoc*: $(\bigcup v \in y \odot_i z. x \odot_i v) = (\bigcup v \in x \odot_i y. v \odot_i z)$

begin

sublocale *ims*: *multisemigroup* $\lambda x\ y.\ x \odot_i y$
 ⟨*proof*⟩

abbreviation $DD \equiv ims.\Delta$

lemma *iconv-prop*: $X \star_i Y \equiv ims.conv\ i\ X\ Y$
 ⟨*proof*⟩

end

class *st-imultimagma* = *imultimagma* +
 fixes *src* :: $nat \Rightarrow 'a \Rightarrow 'a$
 and *tgt* :: $nat \Rightarrow 'a \Rightarrow 'a$
 assumes *Dst*: $x \odot_i y \neq \{\} \Longrightarrow tgt\ i\ x = src\ i\ y$
 and *src-absorb* [*simp*]: $(src\ i\ x) \odot_i x = \{x\}$
 and *tgt-absorb* [*simp*]: $x \odot_i (tgt\ i\ x) = \{x\}$

begin

lemma *inst*: $(src\ 1\ x) \odot_1 x = \{x\}$
 ⟨*proof*⟩

sublocale *stimm*: *st-multimagma* $\lambda x\ y.\ x \odot_i y$ *src i tgt i*
 ⟨*proof*⟩

sublocale *stimm0*: *st-multimagma0* $\lambda x\ y.\ x \odot_i y$ *src i tgt i*
 ⟨*proof*⟩

sublocale *stimm1*: *st-multimagma1* $\lambda x\ y.\ x \odot_i y$ *src i tgt i*
 ⟨*proof*⟩

abbreviation *srcfix* ≡ *stimm.sfix*

abbreviation *tgtfix* ≡ *stimm.tfix*

abbreviation *Srci* $\equiv$ *stimm.Src*

abbreviation *Tgti* $\equiv$ *stimm.Tgt*

end

class *icatoid* = *st-imultimagma* + *imultisemigroup*

sublocale *icatoid* $\subseteq$ *icat*: *catoid* $\lambda x\ y.\ x \odot_i y$ *src i tgt i*
⟨*proof*⟩

class *local-icatoid* = *icatoid* +
assumes *src-local*: $Srci\ i\ (x \odot_i src\ i\ y) \subseteq Srci\ i\ (x \odot_i y)$
and *tgt-local*: $Tgti\ i\ (tgt\ i\ x \odot_i y) \subseteq Tgti\ i\ (x \odot_i y)$

sublocale *local-icatoid* $\subseteq$ *licat*: *local-catoid* $\lambda x\ y.\ x \odot_i y$ *src i tgt i*
⟨*proof*⟩

class *functional-imagma* = *imultimagma* +
assumes *functionality*: $x \in y \odot_i z \Longrightarrow x' \in y \odot_i z \Longrightarrow x = x'$

sublocale *functional-imagma* $\subseteq$ *pmi*: *functional-magma* $\lambda x\ y.\ x \odot_i y$
⟨*proof*⟩

class *functional-isemigroup* = *functional-imagma* + *imultisemigroup*

sublocale *functional-isemigroup* $\subseteq$ *psgi*: *functional-semigroup* $\lambda x\ y.\ x \odot_i y$⟨*proof*⟩

class *functional-icatoid* = *functional-isemigroup* + *icatoid*

sublocale *functional-icatoid* $\subseteq$ *psgi*: *functional-catoid* $\lambda x\ y.\ x \odot_i y$ *src i tgt i*
⟨*proof*⟩

class *icategory* = *functional-icatoid* + *local-icatoid*

sublocale *icategory* $\subseteq$ *icatt*: *single-set-category* $\lambda x\ y.\ x \odot_i y$ *src i tgt i*
⟨*proof*⟩

7.2 ω-Catoids

Next we define ω-catoids and ω-categories.

class *omega-st-multimagma* = *st-imultimagma* +
assumes *comm-sisj*: $i \neq j \Longrightarrow src\ i\ (src\ j\ x) = src\ j\ (src\ i\ x)$
and *comm-sitj*: $i \neq j \Longrightarrow src\ i\ (tgt\ j\ x) = tgt\ j\ (src\ i\ x)$
and *comm-titj*: $i \neq j \Longrightarrow tgt\ i\ \ (tgt\ j\ x) = tgt\ j\ (tgt\ i\ x)$
and *si-hom*: $i \neq j \Longrightarrow Srci\ i\ (x \odot_j y) \subseteq src\ i\ x \odot_j src\ i\ y$
and *ti-hom*: $i \neq j \Longrightarrow Tgti\ i\ (x \odot_j y) \subseteq tgt\ i\ x \odot_j tgt\ i\ y$
and *omega-interchange*: $i < j \Longrightarrow (w \odot_j x) \star_i (y \odot_j z) \subseteq (w \odot_i y) \star_j (x \odot_i z)$

and *sjsi* [*simp*]: $i < j \Longrightarrow src\ j\ (src\ i\ x) = src\ i\ x$
and *sjti* [*simp*]: $i < j \Longrightarrow src\ j\ (tgt\ i\ x) = tgt\ i\ x$
and *tjsi* [*simp*]: $i < j \Longrightarrow tgt\ j\ (src\ i\ x) = src\ i\ x$
and *tjti* [*simp*]: $i < j \Longrightarrow tgt\ j\ (tgt\ i\ x) = tgt\ i\ x$

class *omega-catoid* = *omega-st-multimagma* + *icatoid*

context *omega-st-multimagma*
begin

lemma *omega-interchange-var*: $(w \odot_{(i + k + 1)} x) \star_i (y \odot_{(i + k + 1)} z) \subseteq (w \odot_i y) \star_{(i + k + 1)} (x \odot_i z)$
⟨*proof*⟩

lemma *all-sisj*: $src\ i\ (src\ j\ x) = src\ j\ (src\ i\ x)$
⟨*proof*⟩

lemma *all-titj*: $tgt\ i\ (tgt\ j\ x) = tgt\ j\ (tgt\ i\ x)$
⟨*proof*⟩

lemma *sjsi-var* [*simp*]: $src\ (i + k)\ (src\ i\ x) = src\ i\ x$
⟨*proof*⟩

lemma *sjti-var* [*simp*]: $src\ (i + k)\ (tgt\ i\ x) = tgt\ i\ x$
⟨*proof*⟩

lemma *tjsi-var* [*simp*]: $tgt\ (i + k)\ (src\ i\ x) = src\ i\ x$
⟨*proof*⟩

lemma *tjti-var* [*simp*]: $tgt\ (i + k)\ (tgt\ i\ x) = tgt\ i\ x$
⟨*proof*⟩

The following sublocale statement should help us to translate statements for 2-catoids to ω-catoids. But it does not seem to work. Hence we duplicate the work done for 2-catoids, and later also for semirings and quantales.

sublocale *otmm*: *two-st-multimagma*
$\lambda x\ y.\ x \odot_i y$
$src\ i$
$tgt\ i$
$\lambda x\ y.\ x \odot_{(i + k + 1)} y$
$src\ (i + k + 1)$
$tgt\ (i + k + 1)$
⟨*proof*⟩

end

class *omega-st-multimagma-strong* = *omega-st-multimagma* +
assumes *sj-hom-strong*: $i < j \Longrightarrow Srci\ j\ (x \odot_i y) = src\ j\ x \odot_i src\ j\ y$

and *tj-hom-strong*: $i < j \Longrightarrow Tgti\ j\ (x \odot_i y) = tgt\ j\ x \odot_i tgt\ j\ y$

begin

lemma *sj-hom-strong-var*: $Srci\ (i + k + 1)\ (x \odot_i y) = src\ (i + k + 1)\ x \odot_i src\ (i + k + 1)\ y$
⟨*proof*⟩

lemma *tj-hom-strong-var*: $Tgti\ (i + k + 1)\ (x \odot_i y) = tgt\ (i + k + 1)\ x \odot_i tgt\ (i + k + 1)\ y$
⟨*proof*⟩

end

sublocale *omega-st-multimagma* ⊆ *olropp*: *omega-st-multimagma* $\lambda\ x\ i\ y.\ y \odot_i x$ *tgt src*
⟨*proof*⟩

context *omega-st-multimagma*
begin

lemma *sisj*: $i \leq j \Longrightarrow src\ i\ (src\ j\ x) = src\ i\ x$
⟨*proof*⟩

lemma *sisj-var* [*simp*]: $src\ i\ (src\ (i + k)\ x) = src\ i\ x$
⟨*proof*⟩

lemma *sitj*: $i < j \Longrightarrow src\ i\ (tgt\ j\ x) = src\ i\ x$
⟨*proof*⟩

lemma *sitj-var* [*simp*]: $src\ i\ (tgt\ (i + k + 1)\ x) = src\ i\ x$
⟨*proof*⟩

lemma *tisj*: $i < j \Longrightarrow tgt\ i\ (src\ j\ x) = tgt\ i\ x$
⟨*proof*⟩

lemma *tisj-var* [*simp*]: $tgt\ i\ (src\ (i + k + 1)\ x) = tgt\ i\ x$
⟨*proof*⟩

lemma *titi*: $i \leq j \Longrightarrow tgt\ i\ (tgt\ j\ x) = tgt\ i\ x$
⟨*proof*⟩

lemma *titi-var* [*simp*]: $tgt\ i\ (tgt\ (i + k)\ x) = tgt\ i\ x$
⟨*proof*⟩

end

context *omega-catoid*
begin

lemma *src-icat1*:
  assumes $i \leq j$
  and *DD j x y*
  shows *Srci i* $(x \odot_j y) = \{src\ i\ x\}$
  ⟨*proof*⟩

lemma *src-icat2*:
  assumes $i < j$
  and *DD j x y*
shows *Srci i* $(x \odot_j y) = \{src\ i\ y\}$
  ⟨*proof*⟩

lemma *tgt-icat1*:
  assumes $i < j$
  and *DD j x y*
  shows *Tgti i* $(x \odot_j y) = \{tgt\ i\ x\}$
  ⟨*proof*⟩

lemma *tgt-icat2*:
  assumes $i \leq j$
  and *DD j x y*
  shows *Tgti i* $(x \odot_j y) = \{tgt\ i\ y\}$
  ⟨*proof*⟩

end

We lift the axioms to the powerset level.

context *omega-st-multimagma*
begin

lemma *comm-SiSj*: *Srci i* (*Srci j X*) = *Srci j* (*Srci i X*)
  ⟨*proof*⟩

lemma *comm-TiTj*: *Tgti i* (*Tgti j X*) = *Tgti j* (*Tgti i X*)
  ⟨*proof*⟩

lemma *comm-SiTj*: $i \neq j \Longrightarrow$ *Srci i* (*Tgti j x*) = *Tgti j* (*Srci i x*)
  ⟨*proof*⟩

lemma *comm-TiSj*: $i \neq j \Longrightarrow$ *Tgti i* (*Srci j x*) = *Srci j* (*Tgti i x*)
  ⟨*proof*⟩

lemma *interchange-lift*:
  assumes $i < j$
  shows $(W \star_j X) \star_i (Y \star_j Z) \subseteq (W \star_i Y) \star_j (X \star_i Z)$
⟨*proof*⟩

lemma *Srcj-hom*:

assumes $i \neq j$
shows *Srci j* $(X \star_i Y) \subseteq$ *Srci j X* $\star_i$ *Srci j Y*
⟨*proof*⟩

lemma *Tgtj-hom*:
assumes $i \neq j$
shows *Tgti j* $(X \star_i Y) \subseteq$ *Tgti j X* $\star_i$ *Tgti j Y*
⟨*proof*⟩

lemma *SjSi*: $i \leq j \Longrightarrow$ *Srci j* (*Srci i X*) = *Srci i X*
⟨*proof*⟩

lemma *SjSi-var* [*simp*]: *Srci* $(i + k)$ (*Srci i X*) = *Srci i X*
⟨*proof*⟩

lemma *SjTi*: $i \leq j \Longrightarrow$ *Srci j* (*Tgti i X*) = *Tgti i X*
⟨*proof*⟩

lemma *SjTi-var* [*simp*]: *Srci* $(i + k)$ (*Tgti i X*) = *Tgti i X*
⟨*proof*⟩

lemma *TjSi*: $i \leq j \Longrightarrow$ *Tgti j* (*Srci i X*) = *Srci i X*
⟨*proof*⟩

lemma *TjSi-var* [*simp*]: *Tgti* $(i + k)$ (*Srci i X*) = *Srci i X*
⟨*proof*⟩

lemma *TjTi*: $i \leq j \Longrightarrow$ *Tgti j* (*Tgti i X*) = *Tgti i X*
⟨*proof*⟩

lemma *TjTi-var* [*simp*]: *Tgti* $(i + k)$ (*Tgti i X*) = *Tgti i X*
⟨*proof*⟩

lemma *srcfixij*: $i \leq j \Longrightarrow$ *srcfix i* $\subseteq$ *srcfix i* $\star_j$ *srcfix i*
⟨*proof*⟩

lemma *srcfixij-var*: *srcfix i* $\subseteq$ *srcfix i* $\star_{(j + k)}$ *srcfix i*
⟨*proof*⟩

lemma *srcfixij-var2*: $i \leq j \Longrightarrow$ *srcfix i* $\subseteq$ *srcfix j*
⟨*proof*⟩

lemma *srcfixij-var3*: *srcfix i* $\subseteq$ *srcfix* $(i + k)$
⟨*proof*⟩

end

context *omega-st-multimagma-strong*
begin

lemma *Srcj-hom-strong*:
  assumes $i < j$
  shows $Srci\ j\ (X \star_i Y) = Srci\ j\ X \star_i Srci\ j\ Y$
⟨*proof*⟩

lemma *Srcj-hom-strong-var*: $Srci\ (i + k + 1)\ (X \star_i Y) = Srci\ (i + k + 1)\ X \star_i Srci\ (i + k + 1)\ Y$
  ⟨*proof*⟩

lemma *Tgtj-hom-strong*:
  assumes $i < j$
  shows $Tgti\ j\ (X \star_i Y) = Tgti\ j\ X \star_i Tgti\ j\ Y$
⟨*proof*⟩

lemma *Tgtj-hom-strong-var*: $Tgti\ (i + k + 1)\ (X \star_i Y) = Tgti\ (i + k + 1)\ X \star_i Tgti\ (i + k + 1)\ Y$
  ⟨*proof*⟩

lemma *srcfixij-var2*: $i < j \Longrightarrow srcfix\ j \star_i srcfix\ j \subseteq srcfix\ j$
  ⟨*proof*⟩

lemma *srcfixij-var22*: $srcfix\ (i + k + 1) \star_i srcfix\ (i + k + 1) \subseteq srcfix\ (i + k + 1)$
  ⟨*proof*⟩

lemma *srcfixij-eq*: $i < j \Longrightarrow srcfix\ j \star_i srcfix\ j = srcfix\ j$
  ⟨*proof*⟩

lemma *srcfixij-eq-var* [*simp*]: $srcfix\ (i + k + 1) \star_i srcfix\ (i + k + 1) = srcfix\ (i + k + 1)$
  ⟨*proof*⟩

end

7.3 ω-Catoids and single-set ω-categories

class *omega-catoid-strong* = *omega-st-multimagma-strong* + *omega-catoid*

class *local-omega-catoid* = *omega-st-multimagma* + *local-icatoid*

class *functional-omega-catoid* = *omega-st-multimagma* + *functional-icatoid*

class *local-omega-catoid-strong* = *omega-st-multimagma-strong* + *local-icatoid*

class *omega-category* = *omega-st-multimagma* + *icategory*

begin

lemma *sj-hom-strong*:
  assumes $i < j$
  shows $Srci\ j\ (x \odot_i y) = src\ j\ x \odot_i src\ j\ y$
  ⟨*proof*⟩

lemma *sj-hom-strong-var*: $Srci\ (i + k + 1)\ (x \odot_i y) = src\ (i + k + 1)\ x \odot_i src\ (i + k + 1)\ y$
  ⟨*proof*⟩

lemma *sj-hom-strong-delta*: $i < j \Longrightarrow DD\ i\ x\ y = DD\ i\ (src\ j\ x)\ (src\ j\ y)$
  ⟨*proof*⟩

lemma *tj-hom-strong*: $i < j \Longrightarrow Tgti\ j\ (x \odot_i y) = tgt\ j\ x \odot_i tgt\ j\ y$
  ⟨*proof*⟩

lemma *tj-hom-strong-var*: $Tgti\ (i + k + 1)\ (x \odot_i y) = tgt\ (i + k + 1)\ x \odot_i tgt\ (i + k + 1)\ y$
  ⟨*proof*⟩

lemma *tj-hom-strong-delta*: $i < j \Longrightarrow DD\ i\ x\ y = DD\ i\ (tgt\ j\ x)\ (tgt\ j\ y)$
  ⟨*proof*⟩

lemma *convi-sgl*: $a \in x \odot_i y \Longrightarrow \{a\} = x \odot_i y$
  ⟨*proof*⟩

Next we derive some simple globular properties.

lemma *strong-interchange-STj*:
  assumes $i < j$
  and $a \in (w \odot_i x) \star_j (y \odot_i z)$
  shows $Tgti\ j\ (w \odot_i x) = Srci\ j\ (y \odot_i z)$
  ⟨*proof*⟩

lemma *strong-interchange-ssi*:
  assumes $i < j$
  and $a \in (w \odot_i x) \star_j (y \odot_i z)$
shows $src\ i\ w = src\ i\ y$
  ⟨*proof*⟩

end

7.4 Reduced axiomatisations

class *omega-st-multimagma-red* = *st-imultimagma* +
  assumes *interchange*: $i < j \Longrightarrow (w \odot_j x) \star_i (y \odot_j z) \subseteq (w \odot_i y) \star_j (x \odot_i z)$
  assumes *srcj-hom*: $i < j \Longrightarrow Srci\ j\ (x \odot_i y) = src\ j\ x \odot_i srcj\ y$
  and *tgtj-hom*: $i < j \Longrightarrow Tgti\ j\ (x \odot_i y) = tgt\ j\ x \odot_i tgt\ j\ y$
  and *srci-weak-hom*: $i < j \Longrightarrow Srci\ i\ (x \odot_j y) \subseteq src\ i\ x \odot_j src\ i\ y$
  and *tgti-weak-hom*: $i < j \Longrightarrow Tgti\ i\ (x \odot_j y) \subseteq src\ i\ x \odot_j src\ i\ y$

begin

lemma *sitjsi* [*simp*]: $src\ i\ (tgt\ j\ (src\ i\ x)) = src\ i\ x$
⟨*proof*⟩

lemma *tisjsi* [*simp*]: $tgt\ i\ (src\ j\ (src\ i\ x)) = src\ i\ x$
⟨*proof*⟩

lemma *sjsi*:
assumes $i \leq j$
shows $src\ j\ (src\ i\ x) = src\ i\ x$
⟨*proof*⟩

lemma *sjti*: $i \leq j \Longrightarrow src\ j\ (tgt\ i\ x) = tgt\ i\ x$
⟨*proof*⟩

lemma *tjsi*: $i \leq j \Longrightarrow tgt\ j\ (src\ i\ x) = src\ i\ x$
⟨*proof*⟩

lemma *tjti*: $i \leq j \Longrightarrow tgt\ j\ \ (tgt\ i\ x) = tgt\ i\ x$
⟨*proof*⟩

lemma *comm-sisj*: $i \neq j \Longrightarrow src\ i\ (src\ j\ x) = src\ j\ (src\ i\ x)$
⟨*proof*⟩

lemma *comm-sitj*: $i \neq j \Longrightarrow src\ i\ (tgt\ j\ x) = tgt\ j\ (src\ i\ x)$
⟨*proof*⟩

lemma *comm-titj*: $i \neq j \Longrightarrow tgt\ i\ (tgt\ j\ x) = tgt\ j\ (tgt\ i\ x)$
⟨*proof*⟩

end

class *omega-catoid-red* = *icatoid* +
assumes *interchange*: $i < j \Longrightarrow (w \odot_j x) \star_i (y \odot_j z) \subseteq (w \odot_i y) \star_j (x \odot_i z)$
and *sj-hom*: $i < j \Longrightarrow Srci\ j\ (x \odot_i y) \subseteq src\ j\ x \odot_i src\ j\ y$
and *tj-hom*: $i < j \Longrightarrow Tgti\ j\ (x \odot_i y) \subseteq tgt\ j\ x \odot_i tgt\ j\ y$

begin

lemma *sitjsi*:
assumes $i < j$
shows $src\ i\ (tgt\ j\ (src\ i\ x)) = src\ i\ x$
⟨*proof*⟩

lemma *tisjsi*: $i < j \Longrightarrow tgt\ i\ (src\ j\ (src\ i\ x)) = src\ i\ x$
⟨*proof*⟩

lemma *sjsi*:

assumes $i < j$
shows $src\ j\ (src\ i\ x) = src\ i\ x$
⟨*proof*⟩

lemma *sjti*: $i < j \Longrightarrow src\ j\ (tgt\ i\ x) = tgt\ i\ x$
⟨*proof*⟩

lemma *tjsi*: $i < j \Longrightarrow tgt\ j\ (src\ i\ x) = src\ i\ x$
⟨*proof*⟩

lemma *tjti*: $i < j \Longrightarrow tgt\ j\ \ (tgt\ i\ x) = tgt\ i\ x$
⟨*proof*⟩

lemma *comm-sisj*: $i < j \Longrightarrow src\ i\ (src\ j\ x) = src\ j\ (src\ i\ x)$
⟨*proof*⟩

lemma *comm-sitj*: $i < j \Longrightarrow src\ i\ (tgt\ j\ x) = tgt\ j\ (src\ i\ x)$
⟨*proof*⟩

lemma *comm-tisj*: $i < j \Longrightarrow tgt\ i\ (src\ j\ x) = src\ j\ (tgt\ i\ x)$
⟨*proof*⟩

lemma *comm-titj*: $i < j \Longrightarrow tgt\ i\ (tgt\ j\ x) = tgt\ j\ (tgt\ i\ x)$
⟨*proof*⟩

lemma *si-hom*: $i < j \Longrightarrow Srci\ i\ \ (x \odot_j y) \subseteq src\ i\ x \odot_j src\ i\ y$
⟨*proof*⟩

lemma *ti-hom*: $i < j \Longrightarrow Tgti\ i\ (x \odot_j y) \subseteq tgt\ i\ x \odot_j tgt\ i\ y$
⟨*proof*⟩

end

class *omega-catoid-red-strong* = *icatoid* +
assumes *interchange*: $i < j \Longrightarrow (w \odot_j x) \star_i (y \odot_j z) \subseteq (w \odot_i y) \star_j (x \odot_i z)$
and *sj-hom-strong*: $i \leq j \Longrightarrow Srci\ j\ (x \odot_i y) = src\ j\ x \odot_i src\ j\ y$
and *tj-hom-strong*: $i \leq j \Longrightarrow Tgti\ j\ (x \odot_i y) = tgt\ j\ x \odot_i tgt\ j\ y$

begin

lemma *sitjsi*: $i < j \Longrightarrow src\ i\ (tgt\ j\ (src\ i\ x)) = src\ i\ x$
⟨*proof*⟩

lemma *tisjsi*: $i < j \Longrightarrow tgt\ i\ (src\ j\ (src\ i\ x)) = src\ i\ x$
⟨*proof*⟩

lemma *sjsi*:
assumes $i < j$
shows $src\ j\ (src\ i\ x) = src\ i\ x$

⟨*proof*⟩

lemma *sjti*: $i < j \Longrightarrow src\ j\ (tgt\ i\ x) = tgt\ i\ x$
⟨*proof*⟩

lemma *tjsi*: $i < j \Longrightarrow tgt\ j\ (src\ i\ x) = src\ i\ x$
⟨*proof*⟩

lemma *tjti*: $i < j \Longrightarrow tgt\ j\ (tgt\ i\ x) = tgt\ i\ x$
⟨*proof*⟩

lemma *comm-sisj*: $i < j \Longrightarrow src\ i\ (src\ j\ x) = src\ j\ (src\ i\ x)$
⟨*proof*⟩

lemma *comm-sitj*: $i < j \Longrightarrow src\ i\ (tgt\ j\ x) = tgt\ j\ (src\ i\ x)$
⟨*proof*⟩

lemma *comm-tisj*: $i < j \Longrightarrow tgt\ i\ (src\ j\ x) = src\ j\ (tgt\ i\ x)$
⟨*proof*⟩

lemma *comm-titj*: $i < j \Longrightarrow tgt\ i\ (tgt\ j\ x) = tgt\ j\ (tgt\ i\ x)$
⟨*proof*⟩

lemma *s0-weak-hom*: $i < j \Longrightarrow Srci\ i\ (x \odot_j y) \subseteq src\ i\ x \odot_j src\ i\ y$
⟨*proof*⟩

lemma *t0-weak-hom*: $i < j \Longrightarrow Tgti\ i\ (x \odot_j y) \subseteq tgt\ i\ x \odot_j tgt\ i\ y$
⟨*proof*⟩

end

end

8 ω-Kleene algebras

theory *Omega-Kleene-Algebra*
imports *Quantales-Converse.Modal-Kleene-Algebra-Var*

begin

Here we develop ω-semigroups and ω-Kleene algebras.

8.1 Copies for i-structures

class *icomp-op* =
fixes *icomp* :: *'a* ⇒ *nat* ⇒ *'a* ⇒ *'a* (- ·- - [*70*,*70*,*70*]*70*)

class *iid-op* =
fixes *un* :: *nat* ⇒ *'a*

class *istar-op* =
  fixes *star* :: *nat* ⇒ *'a* ⇒ *'a*

class *idom-op* =
  fixes *do* :: *nat* ⇒ *'a* ⇒ *'a*

class *icod-op* =
  fixes *cd* :: *nat* ⇒ *'a* ⇒ *'a*

class *imonoid-mult* = *icomp-op* + *iid-op* +
  assumes *assoc*: $x \cdot_i (y \cdot_i z) = (x \cdot_i y) \cdot_i z$
  and *comp-unl*: $un\ i \cdot_i x = x$
  and *comp-unr*: $x \cdot_i un\ i = x$

class *idioid* = *imonoid-mult* + *join-semilattice-zero* +
  assumes *distl*: $x \cdot_i (y + z) = x \cdot_i y + x \cdot_i z$
  and *distr*: $(x + y) \cdot_i z = x \cdot_i z + y \cdot_i z$
  and *annil*: $0 \cdot_i x = 0$
  and *annir*: $x \cdot_i 0 = 0$

class *ikleene-algebra* = *idioid* + *istar-op* +
  assumes *star-unfoldl*: $un\ i + x \cdot_i star\ i\ x \leq star\ i\ x$
  and *star-unfoldr*: $un\ i + star\ i\ x \cdot_i x \leq star\ i\ x$
  and *star-inductl*: $z + x \cdot_i y \leq y \Longrightarrow star\ i\ x \cdot_i z \leq y$
  and *star-inductr*: $z + y \cdot_i x \leq y \Longrightarrow z \cdot_i star\ i\ x \leq y$

class *idomain-semiring* = *idioid* + *idom-op* +
  assumes *do-absorb*: $x \leq do\ i\ x \cdot_i x$
  and *do-local*: $do\ i\ (x \cdot_i do\ i\ y) = do\ i\ (x \cdot_i y)$
  and *do-add*: $do\ i\ (x + y) = do\ i\ x + do\ i\ y$
  and *do-subid*: $do\ i\ x \leq un\ i$
  and *do-zero*: $do\ i\ 0 = 0$

class *icodomain-semiring* = *idioid* + *icod-op* +
  assumes *cd-absorb*: $x \leq x \cdot_i cd\ i\ x$
  and *cd-local*: $cd\ i\ (cd\ i\ x \cdot_i y) = cd\ i\ (x \cdot_i y)$
  and *cd-add*: $cd\ i\ (x + y) = cd\ i\ x + cd\ i\ y$
  and *cd-subid*: $cd\ i\ x \leq un\ i$
  and *cd-zero*: $cd\ i\ 0 = 0$

class *imodal-semiring* = *idomain-semiring* + *icodomain-semiring* +
  assumes *dc-compat*: $do\ i\ (cd\ i\ x) = cd\ i\ x$
  and *cd-compat*: $cd\ i\ (do\ i\ x) = do\ i\ x$

class *imodal-kleene-algebra* = *imodal-semiring* + *ikleene-algebra*

sublocale *imonoid-mult* ⊆ *mm*: *monoid-mult un i* $\lambda x\ y.\ x \cdot_i y$
  ⟨*proof*⟩

sublocale *idioid* $\subseteq$ *di*: *dioid-one-zero* - $\lambda x\ y.\ x \cdot_i y$ *un i* - - -
⟨*proof*⟩

sublocale *idioid* $\subseteq$ *ddi*: *idioid* - - - -$\lambda x\ i\ y.$ *icomp y i x* -
⟨*proof*⟩

sublocale *ikleene-algebra* $\subseteq$ *ki*: *kleene-algebra* - $\lambda x\ y.\ x \cdot_i y$ *un i* - - - *star i*
⟨*proof*⟩

sublocale *ikleene-algebra* $\subseteq$ *dki*: *ikleene-algebra* - - - - $\lambda x\ i\ y.\ y \cdot_i x$ - -
⟨*proof*⟩

sublocale *idomain-semiring* $\subseteq$ *dsri*: *domain-semiring* - $\lambda x\ y.\ x \cdot_i y$ *un i* - *do i* - -
⟨*proof*⟩

sublocale *icodomain-semiring* $\subseteq$ *csri*: *range-semiring* - $\lambda x\ y.\ x \cdot_i y$ *un i* - *cd i* - -
⟨*proof*⟩

sublocale *icodomain-semiring* $\subseteq$ *ds0dual*: *idomain-semiring* - - - - $\lambda x\ i\ y.\ y \cdot_i x$ - *cd*
⟨*proof*⟩

sublocale *imodal-semiring* $\subseteq$ *msri*: *dr-modal-semiring* - $\lambda x\ y.\ x \cdot_i y$ *un i* - *do i* - - *cd i*
⟨*proof*⟩

sublocale *imodal-semiring* $\subseteq$ *msridual*: *imodal-semiring do* - - - - $\lambda x\ i\ y.\ y \cdot_i x$ - *cd*
⟨*proof*⟩

sublocale *imodal-kleene-algebra* $\subseteq$ *mkai*: *dr-modal-kleene-algebra* - $\lambda x\ y.\ x \cdot_i y$ *un i* - - - *star i do i cd i*⟨*proof*⟩

sublocale *imodal-kleene-algebra* $\subseteq$ *mkaidual*: *imodal-kleene-algebra* - - - - $\lambda x\ i\ y.\ y \cdot_i x$ - - *do cd* ⟨*proof*⟩

8.2 Globular ω-semirings

class *omega-semiring* = *imodal-semiring* +
assumes *interchange*: $i < j \Longrightarrow (w \cdot_j x) \cdot_i (y \cdot_j z) \leq (w \cdot_i y) \cdot_j (x \cdot_i z)$
and *di-hom*: $i \neq j \Longrightarrow$ *do i* $(x \cdot_j y) \leq$ *do i x* $\cdot_j$ *do i y*
and *ci-hom*: $i \neq j \Longrightarrow$ *cd i* $(x \cdot_j y) \leq$ *cd i x* $\cdot_j$ *cd i y*
and *djdi*: $i < j \Longrightarrow$ *do j* (*do i x*) = *do i x*

class *strong-omega-semiring* = *omega-semiring* +
assumes *dj-strong-hom*: $i < j \Longrightarrow$ *do j* $(x \cdot_i y)$ = *do j x* $\cdot_i$ *do j y*
and *cj-strong-hom*: $i < j \Longrightarrow$ *cd j* $(x \cdot_i y)$ = *cd j x* $\cdot_i$ *cd j y*

sublocale *omega-semiring* ⊆ *tgsdual*: *omega-semiring do - - - -* $\lambda x\ i\ y.\ y \cdot_i x$ *- cd*
⟨*proof*⟩

sublocale *strong-omega-semiring* ⊆ *stgsdual*: *strong-omega-semiring do - - - -* $\lambda x\ i\ y.\ y \cdot_i x$ *- cd*
⟨*proof*⟩

context *omega-semiring*
begin

lemma *interchange-var*: $(w \cdot_{(i + k + 1)} x) \cdot_i (y \cdot_{(i + k + 1)} z) \leq (w \cdot_i y) \cdot_{(i + k + 1)} (x \cdot_i z)$
⟨*proof*⟩

lemma *djdi-var* [*simp*]: $do\ (i + k + 1)\ (do\ i\ x) = do\ i\ x$
⟨*proof*⟩

lemma *cjdi*: $i \leq j \Longrightarrow cd\ j\ (do\ i\ x) = do\ i\ x$
⟨*proof*⟩

lemma *cjdi-var* [*simp*]: $cd\ (i + k)\ (do\ i\ x) = do\ i\ x$
⟨*proof*⟩

lemma *djci*: $i \leq j \Longrightarrow do\ j\ (cd\ i\ x) = cd\ i\ x$
⟨*proof*⟩

lemma *djci-var* [*simp*]: $do\ (i + k)\ (cd\ i\ x) = cd\ i\ x$
⟨*proof*⟩

lemma *cjci*: $i \leq j \Longrightarrow cd\ j\ (cd\ i\ x) = cd\ i\ x$
⟨*proof*⟩

lemma *cjci-var* [*simp*]: $cd\ (i + k)\ (cd\ i\ x) = cd\ i\ x$
⟨*proof*⟩

lemma *unj-compi-var*: $i \leq j \Longrightarrow un\ j \leq un\ j \cdot_i un\ j$
⟨*proof*⟩

lemma *un-iso*: $i \leq j \Longrightarrow un\ i \leq un\ j$
⟨*proof*⟩

lemma *uni-compj-eq* : $i < j \Longrightarrow un\ i \cdot_j un\ i = un\ i$
⟨*proof*⟩

lemma *uni-compj-eq-var* [*simp*]: $un\ i \cdot_{(i + k)} un\ i = un\ i$
⟨*proof*⟩

lemma *dj-uni*: $i < j \Longrightarrow do\ j\ (un\ i) = un\ i$
⟨*proof*⟩

lemma *dj-uni-var* [*simp*]: $do\ (i + k)\ (un\ i) = un\ i$
⟨*proof*⟩

lemma *di-unj*: $i < j \Longrightarrow do\ i\ (un\ j) = un\ i$
⟨*proof*⟩

lemma *di-unj-var* [*simp*]: $do\ i\ (un\ (i + k)) = un\ i$
⟨*proof*⟩

lemma *ci-unj*: $i < j \Longrightarrow cd\ i\ (un\ j) = un\ i$
⟨*proof*⟩

lemma *ci-unj-var* [*simp*]: $cd\ i\ (un\ (i + k)) = un\ i$
⟨*proof*⟩

lemma *cj-uni*: $i < j \Longrightarrow cd\ j\ (un\ i) = un\ i$
⟨*proof*⟩

lemma *cj-uni-var* [*simp*]: $cd\ (i + k)\ (un\ i) = un\ i$
⟨*proof*⟩

lemma *comm-didj*: $i \leq j \Longrightarrow do\ i\ (do\ j\ x) = do\ j\ (do\ i\ x)$
⟨*proof*⟩

lemma *comm-didj-var*: $do\ i\ (do\ (i + k)\ x) = do\ (i + k)\ (do\ i\ x)$
⟨*proof*⟩

lemma *comm-dicj*: $i < j \Longrightarrow do\ i\ (cd\ j\ x) = cd\ j\ (do\ i\ x)$
⟨*proof*⟩

lemma *comm-dicj-var*: $do\ i\ (cd\ (i + k + 1)\ x) = cd\ (i + k + 1)\ (do\ i\ x)$
⟨*proof*⟩

lemma *comm-cicj*: $i \leq j \Longrightarrow cd\ i\ (cd\ j\ x) = cd\ j\ (cd\ i\ x)$
⟨*proof*⟩

lemma *comm-cicj-var* [*simp*]: $cd\ i\ (cd\ (i + k)\ x) = cd\ (i + k)\ (cd\ i\ x)$
⟨*proof*⟩

lemma *comm-cidj*: $i < j \Longrightarrow cd\ i\ (do\ j\ x) = do\ j\ (cd\ i\ x)$
⟨*proof*⟩

We prove further lemmas that are not related to the globular structure.

lemma *di-compi-idem*: $i \leq j \Longrightarrow do\ i\ x \cdot_j do\ i\ x = do\ i\ x$
⟨*proof*⟩

lemma *di-compi-idem-var* [*simp*]: $do\ i\ x \cdot_{(i + k)} do\ i\ x = do\ i\ x$
⟨*proof*⟩

lemma *codi-compj-idem*: $i \leq j \Longrightarrow cd\ i\ x \cdot_j cd\ i\ x = cd\ i\ x$
⟨*proof*⟩

lemma *codi-compj-idem-var* [*simp*]: $cd\ i\ x \cdot_{(i\ +\ k)} cd\ i\ x = cd\ i\ x$
⟨*proof*⟩

lemma *domij-loc*: $i \leq j \Longrightarrow do\ i\ (x \cdot_j do\ j\ y) = do\ i\ (x \cdot_j y)$
⟨*proof*⟩

lemma *domij-loc-var* [*simp*]: $do\ i\ (x \cdot_{(i\ +\ k)} do\ (i + k)\ y) = do\ i\ (x \cdot_{(i\ +\ k)} y)$
⟨*proof*⟩

lemma *codij-locl*: $i \leq j \Longrightarrow cd\ i\ (cd\ j\ x \cdot_j y) = cd\ i\ (x \cdot_j y)$
⟨*proof*⟩

lemma *codij-locl-var* [*simp*]: $cd\ i\ (cd\ (i + k)\ x \cdot_{(i\ +\ k)} y) = cd\ i\ (x \cdot_{(i\ +\ k)} y)$
⟨*proof*⟩

lemma *domij-exp*: $i < j \Longrightarrow do\ i\ (cd\ j\ x \cdot_j y) = do\ i\ (x \cdot_j y)$
⟨*proof*⟩

lemma *domij-exp-var* [*simp*]: $do\ i\ (cd\ (i + k + 1)\ x \cdot_{(i\ +\ k\ +\ 1)} y) = do\ i\ (x \cdot_{(i\ +\ k\ +\ 1)} y)$
⟨*proof*⟩

lemma *codij-exp*: $i < j \Longrightarrow cd\ i\ (x \cdot_j do\ j\ y) = cd\ i\ (x \cdot_j y)$
⟨*proof*⟩

lemma *codij-exp-var* [*simp*]: $cd\ i\ (x \cdot_{(i\ +\ k\ +\ 1)} do\ (i + k + 1)\ y) = cd\ i\ (x \cdot_{(i\ +\ k\ +\ 1)} y)$
⟨*proof*⟩

lemma *domij-loc-var2*: $i \leq j \Longrightarrow do\ i\ (x \cdot_i do\ j\ y) = do\ i\ (x \cdot_i y)$
⟨*proof*⟩

lemma *domij-loc-var3*: $do\ i\ (x \cdot_i do\ (i + k)\ y) = do\ i\ (x \cdot_i y)$
⟨*proof*⟩

lemma *codij-loc-var*: $i \leq j \Longrightarrow cd\ i\ (cd\ j\ x \cdot_i y) = cd\ i\ (x \cdot_i y)$
⟨*proof*⟩

lemma *codij-loc-var2*: $cd\ i\ (cd\ (i + k)\ x \cdot_i y) = cd\ i\ (x \cdot_i y)$
⟨*proof*⟩

lemma *di-compj*: $i < j \Longrightarrow do\ i\ x \cdot_i (y \cdot_j z) \leq (do\ i\ x \cdot_i y) \cdot_j (do\ i\ x \cdot_i z)$
⟨*proof*⟩

lemma *dj-compj*: $i < j \Longrightarrow do\ j\ x \cdot_i (y \cdot_j z) \leq (do\ j\ x \cdot_i y) \cdot_j (do\ j\ x \cdot_i z)$
⟨*proof*⟩

lemma *dij-export*: $i \leq j \Longrightarrow do\ i\ (do\ j\ x \cdot_j y) \leq do\ i\ x \cdot_j do\ i\ y$
⟨*proof*⟩

lemma *dij-export-var* [*simp*]: $do\ i\ (do\ (i + k)\ x \cdot_{(i + k)} y) \leq do\ i\ x \cdot_{(i + k)} do\ i\ y$
⟨*proof*⟩

lemma *codij-export*: $i \leq j \Longrightarrow cd\ i\ (x \cdot_j cd\ j\ y) \leq cd\ i\ x \cdot_j cd\ i\ y$
⟨*proof*⟩

lemma *codij-export-var* [*simp*]: $cd\ i\ (x \cdot_{(i + k)} cd\ (i + k)\ y) \leq cd\ i\ x \cdot_{(i + k)} cd\ i\ y$
⟨*proof*⟩

lemma *dji-export*: $i \leq j \Longrightarrow do\ j\ (do\ i\ x \cdot_j y) = do\ i\ x \cdot_j do\ j\ y$
⟨*proof*⟩

lemma *dji-export-var*: $do\ (i + k)\ (do\ i\ x \cdot_{(i + k)} y) = do\ i\ x \cdot_{(i + k)} do\ (i + k)\ y$
⟨*proof*⟩

lemma *codji-export*: $i \leq j \Longrightarrow cd\ j\ (x \cdot_j cd\ i\ y) = cd\ j\ x \cdot_j cd\ i\ y$
⟨*proof*⟩

lemma *codji-export-var*: $cd\ (i + k)\ (x \cdot_{(i + k)} cd\ i\ y) = cd\ (i + k)\ x \cdot_{(i + k)} cd\ i\ y$
⟨*proof*⟩

lemma *di-compji*: $i \leq j \Longrightarrow do\ i\ x \cdot_j do\ i\ y = do\ i\ x \cdot_i do\ i\ y$
⟨*proof*⟩

lemma *di-compji-var*: $do\ i\ x \cdot_{(i + k)} do\ i\ y = do\ i\ x \cdot_i do\ i\ y$
⟨*proof*⟩

lemma *dom-exchange-strong*: $i \leq j \Longrightarrow (do\ i\ w \cdot_j do\ i\ x) \cdot_i (do\ i\ y \cdot_j do\ i\ z) = (do\ i\ w \cdot_i do\ i\ y) \cdot_j (do\ i\ x \cdot_i do\ i\ z)$
⟨*proof*⟩

lemma *codidomj-exp*: $i < j \Longrightarrow cd\ i\ (x \cdot_i y) \leq cd\ i\ (x \cdot_i cd\ j\ y)$
⟨*proof*⟩

lemma *codidomj-exp-var*: $cd\ i\ (x \cdot_i y) \leq cd\ i\ (x \cdot_i cd\ (i + k + 1)\ y)$
⟨*proof*⟩

The following laws are diamond laws. It remains to define diamonds for them.

lemma *fdiaifdiaj-prop*: $i \leq j \Longrightarrow do\ i\ (y \cdot_i do\ j\ (x \cdot_j z)) = do\ i\ (y \cdot_i (x \cdot_j z))$

⟨*proof*⟩

lemma *bdiaifdiaj-prop*: $i < j \Longrightarrow cd\ i\ (do\ j\ (x \cdot_j z) \cdot_i y) = cd\ i\ ((x \cdot_j z) \cdot_i y)$
⟨*proof*⟩

lemma *fdiaibdiaj-prop*: $i < j \Longrightarrow do\ i\ (y \cdot_i cd\ j\ (x \cdot_j z)) = do\ i\ (y \cdot_i (x \cdot_j z))$
⟨*proof*⟩

lemma *bdiaibdiaj-prop*: $i \leq j \Longrightarrow cd\ i\ (cd\ j\ (x \cdot_j z) \cdot_i y) = cd\ i\ ((x \cdot_j z) \cdot_i y)$
⟨*proof*⟩

lemma *fdiaifdiaj-prop2*: $i < j \Longrightarrow do\ i\ (y \cdot_i do\ j\ (x \cdot_j z)) \leq do\ i\ (y \cdot_i (do\ i\ x \cdot_j do\ i\ z))$
⟨*proof*⟩

lemma *fdiaii-prop2*: $i < j \Longrightarrow do\ i\ (y \cdot_i do\ i\ (x \cdot_j z)) \leq do\ i\ (y \cdot_i (do\ i\ x \cdot_j do\ i\ z))$
⟨*proof*⟩

lemma *bdiaidomj-prop2*: $i < j \Longrightarrow cd\ i\ (do\ j\ (x \cdot_j z) \cdot_i y) \leq cd\ i\ ((cd\ i\ x \cdot_j cd\ i\ z) \cdot_i y)$
⟨*proof*⟩

lemma *bdiaidomi-prop2*: $i < j \Longrightarrow cd\ i\ (do\ i\ (x \cdot_j z) \cdot_i y) \leq cd\ i\ ((do\ i\ x \cdot_j do\ i\ z) \cdot_i y)$
⟨*proof*⟩

lemma *fdiaibdiaj-prop-2*: $i < j \Longrightarrow do\ i\ (y \cdot_i cd\ j\ (z \cdot_j x)) \leq do\ i\ (y \cdot_i (do\ i\ x \cdot_j do\ i\ z))$
⟨*proof*⟩

lemma *fdiaibdiai-prop2*: $i < j \Longrightarrow do\ i\ (y \cdot_i cd\ i\ (z \cdot_j x)) \leq do\ i\ (y \cdot_i (cd\ i\ z \cdot_j cd\ i\ x))$
⟨*proof*⟩

lemma *bdiaibdiaj-prop2*: $i < j \Longrightarrow cd\ i\ (cd\ j\ (z \cdot_j x) \cdot_i y) \leq cd\ i\ ((cd\ i\ x \cdot_j cd\ i\ z) \cdot_i y)$
⟨*proof*⟩

lemma *bdiaibdiai-prop2*: $i < j \Longrightarrow cd\ i\ (cd\ i\ (x \cdot_j z) \cdot_i y) \leq cd\ i\ ((cd\ i\ x \cdot_j cd\ i\ z) \cdot_i y)$
⟨*proof*⟩

lemma *fdiajfdiai-prop3*: $i < j \Longrightarrow do\ j\ (x \cdot_j do\ i\ (y \cdot_i z)) \leq do\ j\ (x \cdot_j do\ i\ (do\ j\ y \cdot_i z))$
⟨*proof*⟩

lemma *bdiajbdiai-prop3*: $i < j \Longrightarrow cd\ j\ (cd\ i\ (z \cdot_i y) \cdot_j x) \leq cd\ j\ (cd\ i\ (z \cdot_i cd\ j\ y) \cdot_j x)$

⟨*proof*⟩

end

The following proofs need the domain codomain duality, which has been formalised using a sublocale statement above. It is only available outside of a context.

lemma (**in** *omega-semiring*) *domicodj-exp*: $i < j \Longrightarrow do\ i\ (x \cdot_i y) \leq do\ i\ (cd\ j\ x \cdot_i y)$
⟨*proof*⟩

lemma (**in** *omega-semiring*) *domicodj-exp-var*: $do\ i\ (x \cdot_i y) \leq do\ i\ (cd\ (i + k + 1)\ x \cdot_i y)$
⟨*proof*⟩

lemma (**in** *omega-semiring*) *fdiajbdiai-prop3*: $i < j \Longrightarrow do\ j\ (x \cdot_j cd\ i\ (z \cdot_i y)) \leq do\ j\ (x \cdot_j cd\ i\ (z \cdot_i do\ j\ y))$
⟨*proof*⟩

lemma (**in** *omega-semiring*) *bdiajfdiai-prop3*: $i < j \Longrightarrow cd\ j\ (do\ i\ (y \cdot_i z) \cdot_j x) \leq cd\ j\ (do\ i\ (cd\ j\ y \cdot_i z) \cdot_j x)$
⟨*proof*⟩

context *strong-omega-semiring*
begin

lemma *idj-compj*: $i \leq j \Longrightarrow un\ j \cdot_i un\ j \leq un\ j$
⟨*proof*⟩

lemma *idj-compi-eq*: $i < j \Longrightarrow un\ j = un\ j \cdot_i un\ j$
⟨*proof*⟩

lemma *domicodj-exp*: $i < j \Longrightarrow do\ i\ (x \cdot_i y) = do\ i\ (cd\ j\ x \cdot_i y)$
⟨*proof*⟩

lemma *domicodj-exp-var* [*simp*] : $do\ i\ (cd\ (i + k + 1)\ x \cdot_i y) = do\ i\ (x \cdot_i y)$
⟨*proof*⟩

lemma *codidomj-exp*: $i < j \Longrightarrow cd\ i\ (x \cdot_i do\ j\ y) = cd\ i\ (x \cdot_i y)$
⟨*proof*⟩

lemma *codidomj-exp-var* [*simp*]: $cd\ i\ (x \cdot_i do\ (i + k + 1)\ y) = cd\ i\ (x \cdot_i y)$
⟨*proof*⟩

lemma *fdiajfdiai-prop3*: $i < j \Longrightarrow do\ j\ (x \cdot_j do\ i\ (do\ j\ y \cdot_i z)) = do\ j\ (x \cdot_j do\ i\ (y \cdot_i z))$
⟨*proof*⟩

lemma *fdiajbdiai-prop3*: $i < j \Longrightarrow do\ j\ (x \cdot_j cd\ i\ (z \cdot_i do\ j\ y)) = do\ j\ (x \cdot_j cd\ i\ (z$

$\cdot_i$ *y*))
⟨*proof*⟩

lemma *bdiajfdiai-prop3*: $i < j \Longrightarrow cd\ j\ (do\ i\ (cd\ j\ y \cdot_i z) \cdot_j x) = cd\ j\ (do\ i\ (y \cdot_i z) \cdot_j x)$
⟨*proof*⟩

lemma *bdiajbdiai-prop3*: $i < j \Longrightarrow cd\ j\ (cd\ i\ (z \cdot_i cd\ j\ y) \cdot_j x) = cd\ j\ (cd\ i\ (z \cdot_i y) \cdot_j x)$
⟨*proof*⟩

lemma *fdiaifdiaj-prop4*: $i < j \Longrightarrow do\ i\ z \cdot_i do\ j\ (x \cdot_j y) \leq do\ j\ ((do\ i\ z \cdot_i x) \cdot_j (do\ i\ z \cdot_i y))$
⟨*proof*⟩

lemma *fdia0bdia1-prop4*: $i < j \Longrightarrow do\ i\ z \cdot_i cd\ j\ (y \cdot_j x) \leq cd\ j\ ((do\ i\ z \cdot_i y) \cdot_j (do\ i\ z \cdot_i x))$
⟨*proof*⟩

lemma *fdiajfdiaj-prop4*: $i < j \Longrightarrow do\ j\ (x \cdot_j y) \cdot_i do\ i\ z \leq do\ j\ ((x \cdot_i do\ i\ z) \cdot_j (y \cdot_i do\ i\ z))$
⟨*proof*⟩

lemma *bdiajbdiaj-prop4*: $i < j \Longrightarrow cd\ j\ (y \cdot_j x) \cdot_i do\ i\ z \leq cd\ j\ ((y \cdot_i do\ i\ z) \cdot_j (x \cdot_i do\ i\ z))$
⟨*proof*⟩

end

8.3 Globular ω-Kleene algebras

class *omega-kleene-algebra* = *omega-semiring* + *ikleene-algebra*

class *strong-omega-kleene-algebra* = *strong-omega-semiring* + *ikleene-algebra*

context *omega-kleene-algebra*
begin

lemma *interchange-var1*: $i < j \Longrightarrow (x \cdot_j x) \cdot_i ((y \cdot_j y) \cdot_i (z \cdot_j z)) \leq (x \cdot_i (y \cdot_i z)) \cdot_j (x \cdot_i (y \cdot_i z))$
⟨*proof*⟩

lemma *interchange-var2*: $i < j \Longrightarrow (x \cdot_j y) \cdot_i ((x \cdot_j y) \cdot_i (x \cdot_j y)) \leq (x \cdot_i (x \cdot_i x)) \cdot_j (y \cdot_i (y \cdot_i y))$
⟨*proof*⟩

lemma *star-compj*:
assumes $i < j$
shows $star\ i\ (x \cdot_j y) \leq star\ i\ x \cdot_j star\ i\ y$

⟨*proof*⟩

lemma *star-compj-var*: $star\ i\ (x \cdot_{(i\ +\ k\ +\ 1)}\ y) \leq star\ i\ x \cdot_{(i\ +\ k\ +\ 1)}\ star\ i\ y$
⟨*proof*⟩

end

end

9 ω-Quantales

theory *Omega-Quantale*
imports *Quantales-Converse.Modal-Quantale Omega-Kleene-Algebra*

begin

class *iquantale* = *complete-lattice* + *imonoid-mult* +
assumes *Sup-distl*: $x \cdot_i \bigsqcup Y = (\bigsqcup y \in Y.\ x \cdot_i y)$
assumes *Sup-distr*: $\bigsqcup X \cdot_i y = (\bigsqcup x \in X.\ x \cdot_i y)$

sublocale *iquantale* ⊆ *qiq*: *unital-quantale un i* $\lambda x\ y.\ x \cdot_i y$ - - - - - - - -
⟨*proof*⟩

definition (**in** *iquantale*) *istar* = *qiq.qstar*

lemma (**in** *iquantale*) *istar-unfold*: $istar\ i\ x = (\bigsqcup k.\ mm.power\ i\ x\ k)$
⟨*proof*⟩

sublocale *iquantale* ⊆ *dqisi*: *idioid* (⊔) (≤) (<) ⊥ *icomp un*
⟨*proof*⟩

sublocale *iquantale* ⊆ *dqikai*: *ikleene-algebra* (⊔) (≤) (<) ⊥ *icomp un istar*
⟨*proof*⟩

class *idomain-quantale* = *iquantale* + *idom-op* +
assumes *do-absorb*: $x \leq do\ i\ x \cdot_i x$
and *do-local* [*simp*]: $do\ i\ (x \cdot_i do\ i\ y) = do\ i\ (x \cdot_i y)$
and *do-add*: $do\ i\ (x \sqcup y) = do\ i\ x \sqcup do\ i\ y$
and *do-subid*: $do\ i\ x \leq un\ i$
and *do-zero* [*simp*]: $do\ i\ \bot = \bot$

sublocale *idomain-quantale* ⊆ *dqidq*: *domain-quantale do i un i* $\lambda x\ y.\ x \cdot_i y$ - - -
- - - - -
⟨*proof*⟩

sublocale *idomain-quantale* ⊆ *dqidsi*: *idomain-semiring* (⊔) (≤) (<) ⊥ *icomp un do*
⟨*proof*⟩

class *icodomain-quantale* = *iquantale* + *icod-op* +
  assumes *cd-absorb*: $x \leq x \cdot_i cd\ i\ x$
  and *cd-local* [*simp*]: $cd\ i\ (cd\ i\ x \cdot_i y) = cd\ i\ (x \cdot_i y)$
  and *cd-add*: $cd\ i\ (x \sqcup y) = cd\ i\ x \sqcup cd\ i\ y$
  and *cd-subid*: $cd\ i\ x \leq un\ i$
  and *cd-zero* [*simp*]: $cd\ i\ \bot = \bot$

sublocale *icodomain-quantale* ⊆ *cdqicdq*: *codomain-quantale un i* $\lambda x\ y.\ x \cdot_i y$ - - - - - - - - *cd i*
  ⟨*proof*⟩

sublocale *icodomain-quantale* ⊆ *cdqidcsi*: *icodomain-semiring cd* (⊔) (≤) (<) ⊥ *icomp un*
  ⟨*proof*⟩

class *imodal-quantale* = *idomain-quantale* + *icodomain-quantale* +
  assumes *dc-compat* [*simp*]: $do\ i\ (cd\ i\ x) = cd\ i\ x$
  and *cd-compat* [*simp*]: $cd\ i\ (do\ i\ x) = do\ i\ x$

sublocale *imodal-quantale* ⊆ *mqimq*: *dc-modal-quantale un i* $\lambda x\ y.\ x \cdot_i y$ - - - - - - - - *cd i do i*
  ⟨*proof*⟩

sublocale *imodal-quantale* ⊆ *mqimka*: *imodal-kleene-algebra* (⊔) (≤) (<) ⊥ *icomp un istar cd do*
  ⟨*proof*⟩

sublocale *imodal-quantale* ⊆ *mqidual*: *imodal-quantale do* - - - - - - - - $\lambda x\ i\ y.\ y \cdot_i x$ - *cd*
  ⟨*proof*⟩

class *omega-quantale* = *imodal-quantale* +
  assumes *interchange*: $i < j \Longrightarrow (w \cdot_j x) \cdot_i (y \cdot_j z) \leq (w \cdot_i y) \cdot_j (x \cdot_i z)$
  and *dj-hom*: $i \neq j \Longrightarrow do\ j\ (x \cdot_i y) \leq do\ j\ x \cdot_i do\ j\ y$
  and *cj-hom*: $i \neq j \Longrightarrow cd\ j\ (x \cdot_i y) \leq cd\ j\ x \cdot_i cd\ j\ y$
  and *djdi*: $i < j \Longrightarrow do\ j\ (do\ i\ x) = do\ i\ x$

class *strong-omega-quantale* = *omega-quantale* +
  assumes *dj-strong-hom*: $i < j \Longrightarrow do\ j\ (x \cdot_i y) = do\ j\ x \cdot_i do\ j\ y$
  and *cj-strong-hom*: $i < j \Longrightarrow cd\ j\ (x \cdot_i y) = cd\ j\ x \cdot_i cd\ j\ y$

sublocale *omega-quantale* ⊆ *tgqs*: *omega-semiring cd* (⊔) (≤) (<) ⊥ *icomp un do*
  ⟨*proof*⟩

sublocale *strong-omega-quantale* ⊆ *stgqs*: *strong-omega-semiring cd* (⊔) (≤) (<) ⊥ *icomp un do*
  ⟨*proof*⟩

sublocale *omega-quantale* ⊆ *tgqs*: *omega-kleene-algebra* (⊔) (≤) (<) ⊥ *icomp un* *istar cd do* ⟨*proof*⟩

sublocale *strong-omega-quantale* ⊆ *tgqs*: *strong-omega-kleene-algebra* (⊔) (≤) (<) ⊥ *icomp un* *istar cd do* ⟨*proof*⟩

context *omega-quantale*
begin

lemma *istar-aux*: $i < j \Longrightarrow mm.power\ i\ (x \cdot_j y)\ k \leq mm.power\ i\ x\ k \cdot_j mm.power\ i\ y\ k$
⟨*proof*⟩

lemma *istar-oplax*: $i < j \Longrightarrow istar\ i\ (x \cdot_j y) \leq istar\ i\ x \cdot_j istar\ i\ y$
⟨*proof*⟩

lemma *istar-distli*: $i < j \Longrightarrow x \cdot_i (istar\ j\ y) = (\bigsqcup k.\ x \cdot_i (mm.power\ j\ y\ k))$
⟨*proof*⟩

lemma *istar-distri*: $i < j \Longrightarrow (istar\ j\ x) \cdot_i y = (\bigsqcup k.\ mm.power\ j\ x\ k \cdot_i y)$
⟨*proof*⟩

lemma *istar-distlj*: $i < j \Longrightarrow x \cdot_j (istar\ i\ y) = (\bigsqcup k.\ x \cdot_j (mm.power\ i\ y\ k))$
⟨*proof*⟩

lemma *istar-distrj*: $i < j \Longrightarrow (istar\ i\ x) \cdot_j y = (\bigsqcup k.\ mm.power\ i\ x\ k \cdot_j y)$
⟨*proof*⟩

lemma *istar-laxl-aux-var*: $i < j \Longrightarrow do\ i\ x \cdot_i mm.power\ j\ y\ k \leq mm.power\ j\ (do\ i\ x \cdot_i y)\ k$
⟨*proof*⟩

lemma *istar-laxl-var*:
assumes $i < j$
shows $do\ i\ x \cdot_i istar\ j\ y \leq istar\ j\ (do\ i\ x \cdot_i y)$
⟨*proof*⟩

lemma *istar-laxl-var2*: $do\ i\ x \cdot_i istar\ (i + k + 1)\ y \leq istar\ (i + k + 1)\ (do\ i\ x \cdot_i y)$
⟨*proof*⟩

lemma *istar-laxr-aux-var*: $i < j \Longrightarrow mm.power\ j\ x\ k \cdot_i cd\ i\ y \leq mm.power\ j\ (x \cdot_i cd\ i\ y)\ k$
⟨*proof*⟩

lemma *istar-laxr-var*:
assumes $i < j$
shows $istar\ j\ x \cdot_i cd\ i\ y \leq istar\ j\ (x \cdot_i cd\ i\ y)$
⟨*proof*⟩

lemma *istar-laxr-var2*: *istar* $(i + k + 1)$ $x \cdot_i$ *cd* i $y \leq$ *istar* $(i + k + 1)$ $(x \cdot_i$ *cd* i $y)$
⟨*proof*⟩

lemma *istar-prop*:
assumes $i < j$
shows *istar* j $x \cdot_i$ *istar* j $y = (\bigsqcup k\ l.$ *mm.power* j x $k \cdot_i$ *mm.power* j y $l)$
⟨*proof*⟩

end

context *strong-omega-quantale*
begin

lemma *istar-laxl-aux*: $i < j \Longrightarrow$ *do* j $x \cdot_i$ *mm.power* j y $k \leq$ *mm.power* j (*do* j $x \cdot_i$ $y)$ k
⟨*proof*⟩

lemma *istar-laxl*:
assumes $i < j$
shows *do* j $x \cdot_i$ *istar* j $y \leq$ *istar* j (*do* j $x \cdot_i$ y)
⟨*proof*⟩

lemma *istar-laxr-aux*: $i < j \Longrightarrow$ *mm.power* j x $k \cdot_i$ *cd* j $y \leq$ *mm.power* j $(x \cdot_i$ *cd* j $y)$ k
⟨*proof*⟩

lemma *iqstar-laxr*:
assumes $i < j$
shows *istar* j $x \cdot_i$ *cd* j $y \leq$ *istar* j $(x \cdot_i$ *cd* j $y)$
⟨*proof*⟩

lemma *qstar1-aux*: $i < j \Longrightarrow$ *mm.power* j x $k \cdot_i$ *mm.power* j y $k \leq$ *mm.power* j $(x \cdot_i y)$ k
⟨*proof*⟩

end

end

10 Lifting ω-catoids to powerset ω-quantales

theory *Omega-Catoid-Lifting*
imports *Omega-Catoid Omega-Quantale*

begin

instantiation *set* :: (*local-omega-catoid*) *omega-quantale*

begin

definition *do-set* :: *nat* ⇒ *'a set* ⇒ *'a set* **where**
do i X = Srci i X

definition *cd-set* :: *nat* ⇒ *'a set* ⇒ *'a set* **where**
cd i X = Tgti i X

definition *icomp-set* :: *'a set* ⇒ *nat* ⇒ *'a set* ⇒ *'a set* **where**
$X \cdot_i Y = X \star_i Y$

definition *un-set* :: *nat* ⇒ *'a set* **where**
un i = srcfix i

instance
⟨*proof*⟩

end

end

References

[1] C. Calk, E. Goubault, P. Malbos, and G. Struth. Algebraic coherent confluence and higher globular Kleene algebras. *Log. Methods Comput. Sci.*, 18(4), 2022.

[2] C. Calk, P. Malbos, D. Pous, and G. Struth. Higher catoids, higher quantales and their correspondences. *CoRR*, abs/2307.09253, 2023.

[3] C. Calk and G. Struth. Modal quantales, involutive quantales, Dedekind quantales. *Archive of Formal Proofs*, July 2023. https://isa-afp.org/entries/Quantales_Converse.html, Formal proof development.

[4] G. Struth. Quantales. *Archive of Formal Proofs*, December 2018. https://isa-afp.org/entries/Quantales.html, Formal proof development.

[5] G. Struth. Catoids, categories, groupoids. *Archive of Formal Proofs*, August 2023. https://isa-afp.org/entries/Catoids.html, Formal proof development.