Binary Multirelations

Hitoshi Furusawa and Georg Struth

June 20, 2024

Abstract

Binary multirelations associate elements of a set with its subsets; hence they are binary relations of type $A \times 2^A$. Applications include alternating automata, models and logics for games, program semantics with dual demonic and angelic nondeterministic choices and concurrent dynamic logics. This proof document supports an arXiv article that formalises the basic algebra of multirelations and proposes axiom systems for them, ranging from weak bi-monoids to weak bi-quantales.

Contents

1 Introduction

This proof document contains the formal proofs for an article on *Taming Multirelations* [2]. Individual cross-references to statements in [2] have been added to this document so that both can be read in parallel. The first part of this document contains algebraic axiom systems and equational proofs. Some of these proofs are presented in a human-readable style to indicate the kind of algebraic reasoning involved. The second part contains set-theoretic reasoning with concrete multirelations. Its main purpose is to justify the algebraic development and to prepare the soundness proofs of the algebraic axiomatisations with respect to the concrete multirelational model. Set-theoretic reasoning with multirelations tends to be very tedious and showing detailed proofs has not been the aim.

The algebras of multirelations proposed are based on Peleg's multirelational semantics for concurrent dynamic logic [3]. The most basic axiom systems consider multirelations under the operations of sequential and concurrent composition with two corresponding units. These are enriched by lattice operations and various fixpoints. A main source of complexity is the set-theoretic definition of sequential composition of multirelations, which is based on higher-order logic. Its use often requires the Axiom of Choice. In addition, sequential composition is not associative.

Part of this formalisation is also relevant to a previous approach to concurrent dynamic algebra by Furusawa and Struth [1]. More material on variants of multirelations, game algebras and concurrent dynamic algebras will be added in the future.

The authors are indebted to Alasdair Armstrong and Victor Gomes for help with some tricky Isabelle proofs.

2 C-Algebras

theory *C-Algebras*
imports *Kleene-Algebra.Dioid*
begin

no-notation
times (**infixl** · *70*)

2.1 C-Monoids

We start with the c-monoid axioms. These can be found in Section 4 of [2].

class *proto-monoid* =

fixes *s-id* :: $'a$ (1_σ)
and *s-prod* :: $'a \Rightarrow 'a \Rightarrow 'a$ (**infixl** $\cdot$ *80*)
assumes *s-prod-idl* [*simp*]: $1_\sigma \cdot x = x$
and *s-prod-idr* [*simp*]: $x \cdot 1_\sigma = x$

class *proto-bi-monoid* = *proto-monoid* +
fixes *c-id* :: $'a$ (1_π)
and *c-prod* :: $'a \Rightarrow 'a \Rightarrow 'a$ (**infixl** $\parallel$ *80*)
assumes *c-prod-idl* [*simp*]: $1_\pi \parallel x = x$
and *c-prod-assoc*: $(x \parallel y) \parallel z = x \parallel (y \parallel z)$
and *c-prod-comm*: $x \parallel y = y \parallel x$

class *c-monoid* = *proto-bi-monoid* +
assumes *c1* [*simp*]: $(x \cdot 1_\pi) \parallel x = x$
and *c2* [*simp*]: $((x \cdot 1_\pi) \parallel 1_\sigma) \cdot y = (x \cdot 1_\pi) \parallel y$
and *c3*: $(x \parallel y) \cdot 1_\pi = (x \cdot 1_\pi) \parallel (y \cdot 1_\pi)$
and *c4*: $(x \cdot y) \cdot 1_\pi = x \cdot (y \cdot 1_\pi)$
and *c5* [*simp*]: $1_\sigma \parallel 1_\sigma = 1_\sigma$

begin

Next we define domain explicitly as at the beginning of Section 4 in [2] and start proving the algebraic facts from Section 4. Those involving concrete multirelations, such as Proposition 4.1, are considered in the theory file for multirelations.

definition (**in** *c-monoid*) $d :: 'a \Rightarrow 'a$ **where**
$d\ x = (x \cdot 1_\pi) \parallel 1_\sigma$

lemma *c-prod-idr* [*simp*]: $x \parallel 1_\pi = x$
⟨*proof*⟩

We prove the retraction properties of Lemma 4.2.

lemma *c-idem* [*simp*]: $1_\pi \cdot 1_\pi = 1_\pi$
⟨*proof*⟩

lemma *d-idem* [*simp*]: $d\ (d\ x) = d\ x$
⟨*proof*⟩

lemma *p-id-idem*: $(x \cdot 1_\pi) \cdot 1_\pi = x \cdot 1_\pi$
⟨*proof*⟩

Lemma 4.3.

lemma *c2-d*: $d\ x \cdot y = (x \cdot 1_\pi) \parallel y$
⟨*proof*⟩

lemma *cd-2-var*: $d\ (x \cdot 1_\pi) \cdot y = (x \cdot 1_\pi) \parallel y$
⟨*proof*⟩

lemma *dc-prop1* [*simp*]: $d\ x \cdot 1_\pi = x \cdot 1_\pi$
⟨*proof*⟩

lemma *dc-prop2* [*simp*]: $d\ (x \cdot 1_\pi) = d\ x$
⟨*proof*⟩

lemma *ds-prop* [*simp*]: $d\ x \parallel 1_\sigma = d\ x$
⟨*proof*⟩

lemma *dc* [*simp*]: $d\ 1_\pi = 1_\sigma$
⟨*proof*⟩

Part (5) of this Lemma has already been verified above. The next two statements verify the two algebraic properties mentioned in the proof of Proposition 4.4.

lemma *dc-iso* [*simp*]: $d\ (d\ x \cdot 1_\pi) = d\ x$
⟨*proof*⟩

lemma *cd-iso* [*simp*]: $d\ (x \cdot 1_\pi) \cdot 1_\pi = x \cdot 1_\pi$
⟨*proof*⟩

Proposition 4.5.

lemma *d-conc6*: $d\ (x \parallel y) = d\ x \parallel d\ y$
⟨*proof*⟩

lemma *d-conc-s-prod-ax*: $d\ x \parallel d\ y = d\ x \cdot d\ y$
⟨*proof*⟩

lemma *d-rest-ax* [*simp*]: $d\ x \cdot x = x$
⟨*proof*⟩

lemma *d-loc-ax* [*simp*]: $d\ (x \cdot d\ y) = d\ (x \cdot y)$
⟨*proof*⟩

lemma *d-exp-ax* [*simp*]: $d\ (d\ x \cdot y) = d\ x \cdot d\ y$
⟨*proof*⟩

lemma *d-comm-ax*: $d\ x \cdot d\ y = d\ y \cdot d\ x$
⟨*proof*⟩

lemma *d-s-id-prop* [*simp*]: $d\ 1_\sigma = 1_\sigma$
⟨*proof*⟩

Next we verify the conditions of Proposition 4.6.

lemma *d-s-prod-closed* [*simp*]: $d\ (d\ x \cdot d\ y) = d\ x \cdot d\ y$
⟨*proof*⟩

lemma *d-p-prod-closed* [*simp*]: $d\ (d\ x \parallel d\ y) = d\ x \parallel d\ y$

⟨*proof*⟩

lemma *d-idem2* [*simp*]: $d\ x \cdot d\ x = d\ x$
⟨*proof*⟩

lemma *d-assoc*: $(d\ x \cdot d\ y) \cdot d\ z = d\ x \cdot (d\ y \cdot d\ z)$
⟨*proof*⟩

lemma *iso-1* [*simp*]: $(d\ x \cdot 1_\pi) \parallel 1_\sigma = d\ x$
⟨*proof*⟩

Lemma 4.7.

lemma *x-c-par-idem* [*simp*]: $(x \cdot 1_\pi) \parallel (x \cdot 1_\pi) = x \cdot 1_\pi$
⟨*proof*⟩

lemma *d-idem-par* [*simp*]: $d\ x \parallel d\ x = d\ x$
⟨*proof*⟩

lemma *d-inter-r*: $d\ x \cdot (y \parallel z) = (d\ x \cdot y) \parallel (d\ x \cdot z)$
⟨*proof*⟩

Now we provide the counterexamples of Lemma 4.8.

lemma $(x \parallel y) \cdot d\ z = (x \cdot d\ z) \parallel (y \cdot d\ z)$
nitpick
⟨*proof*⟩

lemma $(x \cdot y) \cdot d\ z = x \cdot (y \cdot d\ z)$
nitpick
⟨*proof*⟩

lemma $1_\pi \cdot x = 1_\pi$
nitpick
⟨*proof*⟩

end

2.2 C-Trioids

We can now define the class of c-trioids and prove properties in this class. This covers the algebraic material of Section 5 in [2].

class *proto-dioid* = *join-semilattice-zero* + *proto-monoid* +
assumes *s-prod-distr*: $(x + y) \cdot z = x \cdot z + y \cdot z$
and *s-prod-subdistl*: $x \cdot y + x \cdot z \leq x \cdot (y + z)$
and *s-prod-annil* [*simp*]: $0 \cdot x = 0$

begin

lemma *s-prod-isol*: $x \leq y \Longrightarrow z \cdot x \leq z \cdot y$

⟨*proof*⟩

lemma *s-prod-isor*: $x \leq y \Longrightarrow x \cdot z \leq y \cdot z$
⟨*proof*⟩

end

class *proto-trioid* = *proto-dioid* + *proto-bi-monoid* +
assumes *p-prod-distl*: $x \parallel (y + z) = x \parallel y + x \parallel z$
and *p-rpd-annir* [*simp*]: $x \parallel 0 = 0$

sublocale *proto-trioid* ⊆ *ab-semigroup-mult c-prod*
⟨*proof*⟩

sublocale *proto-trioid* ⊆ *dioid-one-zero* (+) (∥) 1_π *0* (≤) (<)
⟨*proof*⟩

class *c-trioid* = *proto-trioid* + *c-monoid* +
assumes *c6*: $x \cdot 1_\pi \leq 1_\pi$

begin

We show that every c-trioid is a c-monoid.

subclass *c-monoid* ⟨*proof*⟩

subclass *proto-trioid* ⟨*proof*⟩

lemma $1_\pi \cdot 0 = 1_\pi$
nitpick
⟨*proof*⟩

lemma *zero-p-id-prop* [*simp*]: $(x \cdot 0) \cdot 1_\pi = x \cdot 0$
⟨*proof*⟩

The following facts prove and refute properties related to sequential and parallel subidentities.

lemma *d-subid*: $d\ x = x \Longrightarrow x \leq 1_\sigma$
⟨*proof*⟩

lemma $x \leq 1_\sigma \Longrightarrow d\ x = x$
nitpick
⟨*proof*⟩

lemma *p-id-term*: $x \cdot 1_\pi = x \Longrightarrow x \leq 1_\pi$
⟨*proof*⟩

lemma $x \leq 1_\pi \Longrightarrow x \cdot 1_\pi = x$
nitpick
⟨*proof*⟩

Proposition 5.1. is covered by the theory file on multirelations. We verify the remaining conditions in Proposition 5.2.

lemma *dlp-ax*: $x \leq d\ x \cdot x$
⟨*proof*⟩

lemma *d-add-ax*: $d\ (x + y) = d\ x + d\ y$
⟨*proof*⟩

lemma *d-sub-id-ax*: $d\ x \leq 1_\sigma$
⟨*proof*⟩

lemma *d-zero-ax* [*simp*]: $d\ 0 = 0$
⟨*proof*⟩

We verify the algebraic conditions in Proposition 5.3.

lemma *d-absorb1* [*simp*]: $d\ x + (d\ x \cdot d\ y) = d\ x$
⟨*proof*⟩

lemma *d-absorb2* [*simp*]: $d\ x \cdot (d\ x + d\ y) = d\ x$
⟨*proof*⟩

lemma *d-dist1*: $d\ x \cdot (d\ y + d\ z) = d\ x \cdot d\ y + d\ x \cdot d\ z$
⟨*proof*⟩

lemma *d-dist2*: $d\ x + (d\ y \cdot d\ z) = (d\ x + d\ y) \cdot (d\ x + d\ z)$
⟨*proof*⟩

lemma *d-add-prod-closed* [*simp*]: $d\ (d\ x + d\ y) = d\ x + d\ y$
⟨*proof*⟩

The following properties are not covered in the article.

lemma *x-zero-prop*: $(x \cdot 0) \parallel y = d\ (x \cdot 0) \cdot y$
⟨*proof*⟩

lemma *cda-add-ax*: $d\ ((x + y) \cdot z) = d\ (x \cdot z) + d\ (y \cdot z)$
⟨*proof*⟩

lemma *d-x-zero*: $d\ (x \cdot 0) = (x \cdot 0) \parallel 1_\sigma$
⟨*proof*⟩

Lemma 5.4 is verified below because its proofs are simplified by using facts from the next subsection.

2.3 Results for Concurrent Dynamic Algebra

The following proofs and refutation are related to Section 6 in [2]. We do not consider those involving Kleene algebras in this section. We also do not introduce specific notation for diamond operators.

First we prove Lemma 6.1. Part (1) and (3) have already been verified above. Part (2) and (4) require additional assumptions which are present in the context of concurrent dynamic algebra [1]. We also present the counterexamples from Lemma 6.3.

lemma $(x \cdot y) \cdot d\ z = x \cdot (y \cdot d\ z)$
nitpick
⟨*proof*⟩

lemma $d((x \cdot y) \cdot z) = d\ (x \cdot d\ (y \cdot z))$
nitpick
⟨*proof*⟩

lemma *cda-ax1*: $(x \cdot y) \cdot d\ z = x \cdot (y \cdot d\ z) \Longrightarrow d((x \cdot y) \cdot z) = d\ (x \cdot d\ (y \cdot z))$
⟨*proof*⟩

lemma *d-inter*: $(x \parallel y) \cdot d\ z = (x \cdot d\ z) \parallel (y \cdot d\ z)$
nitpick
⟨*proof*⟩

lemma $d\ ((x \parallel y) \cdot z) = d\ (x \cdot z) \cdot d\ (y \cdot z)$
nitpick
⟨*proof*⟩

lemma *cda-ax2*:
assumes $(x \parallel y) \cdot d\ z = (x \cdot d\ z) \parallel (y \cdot d\ z)$
shows $d\ ((x \parallel y) \cdot z) = d\ (x \cdot z) \cdot d\ (y \cdot z)$
⟨*proof*⟩

Next we present some results that do not feature in the article.

lemma $(x \cdot y) \cdot 0 = x \cdot (y \cdot 0)$
nitpick
⟨*proof*⟩

lemma *d-x-zero-prop* [*simp*]: $d\ (x \cdot 0) \cdot 1_\pi = x \cdot 0$
⟨*proof*⟩

lemma $x \leq 1_\sigma \wedge y \leq 1_\sigma \longrightarrow x \cdot y = x \parallel y$
nitpick
⟨*proof*⟩

lemma $x \cdot (y \parallel z) \leq (x \cdot y) \parallel (x \cdot z)$
nitpick
⟨*proof*⟩

lemma $x \leq x \parallel x$
nitpick
⟨*proof*⟩

Lemma 5.4

lemma *d-lb1*: $d\ x \cdot d\ y \leq d\ x$
⟨*proof*⟩

lemma *d-lb2*: $d\ x \cdot d\ y \leq d\ y$
⟨*proof*⟩

lemma *d-glb*: $d\ z \leq d\ x \wedge d\ z \leq d\ y \Longrightarrow d\ z \leq d\ x \cdot d\ y$
⟨*proof*⟩

lemma *d-glb-iff*: $d\ z \leq d\ x \wedge d\ z \leq d\ y \longleftrightarrow d\ z \leq d\ x \cdot d\ y$
⟨*proof*⟩

lemma *x-zero-le-c*: $x \cdot 0 \leq 1_\pi$
⟨*proof*⟩

lemma *p-subid-lb1*: $(x \cdot 0) \parallel (y \cdot 0) \leq x \cdot 0$
⟨*proof*⟩

lemma *p-subid-lb2*: $(x \cdot 0) \parallel (y \cdot 0) \leq y \cdot 0$
⟨*proof*⟩

lemma *p-subid-idem* [*simp*]: $(x \cdot 0) \parallel (x \cdot 0) = x \cdot 0$
⟨*proof*⟩

lemma *p-subid-glb*: $z \cdot 0 \leq x \cdot 0 \wedge z \cdot 0 \leq y \cdot 0 \Longrightarrow z \cdot 0 \leq (x \cdot 0) \parallel (y \cdot 0)$
⟨*proof*⟩

lemma *p-subid-glb-iff*: $z \cdot 0 \leq x \cdot 0 \wedge z \cdot 0 \leq y \cdot 0 \longleftrightarrow z \cdot 0 \leq (x \cdot 0) \parallel (y \cdot 0)$
⟨*proof*⟩

lemma *x-c-glb*: $z \cdot 1_\pi \leq x \cdot 1_\pi \wedge z \cdot 1_\pi \leq y \cdot 1_\pi \Longrightarrow z \cdot 1_\pi \leq (x \cdot 1_\pi) \parallel (y \cdot 1_\pi)$
⟨*proof*⟩

lemma *x-c-lb1*: $(x \cdot 1_\pi) \parallel (y \cdot 1_\pi) \leq x \cdot 1_\pi$
⟨*proof*⟩

lemma *x-c-lb2*: $(x \cdot 1_\pi) \parallel (y \cdot 1_\pi) \leq y \cdot 1_\pi$
⟨*proof*⟩

lemma *x-c-glb-iff*: $z \cdot 1_\pi \leq x \cdot 1_\pi \wedge z \cdot 1_\pi \leq y \cdot 1_\pi \longleftrightarrow z \cdot 1_\pi \leq (x \cdot 1_\pi) \parallel (y \cdot 1_\pi)$
⟨*proof*⟩

end

2.4 C-Lattices

We can now define c-lattices and prove the results from Section 7 in [2].

class *pbl-monoid* = *proto-trioid* +
  fixes $U :: 'a$
  fixes *meet* :: $'a \Rightarrow 'a \Rightarrow 'a$ (**infixl** $\sqcap$ *70*)
  assumes *U-def*: $x \leq U$
  and *meet-assoc*: $(x \sqcap y) \sqcap z = x \sqcap (y \sqcap z)$
  and *meet-comm*: $x \sqcap y = y \sqcap x$
  and *meet-idem* [*simp*]: $x \sqcap x = x$
  and *absorp1*: $x \sqcap (x + y) = x$
  and *absorp2*: $x + (x \sqcap y) = x$

begin

sublocale *lattice* $(\sqcap)$ $(\leq)$ $(<)$ $(+)$
⟨*proof*⟩

lemma *meet-glb*: $z \leq x \wedge z \leq y \Longrightarrow z \leq x \sqcap y$
  ⟨*proof*⟩

lemma *meet-prop*: $z \leq x \wedge z \leq y \longleftrightarrow z \leq x \sqcap y$
  ⟨*proof*⟩

end

class *pbdl-monoid* = *pbl-monoid* +
  assumes *lat-dist1*: $x + (y \sqcap z) = (x + y) \sqcap (x + z)$

begin

lemma *lat-dist2*: $(x \sqcap y) + z = (x + z) \sqcap (y + z)$
  ⟨*proof*⟩

lemma *lat-dist3*: $x \sqcap (y + z) = (x \sqcap y) + (x \sqcap z)$
⟨*proof*⟩

lemma *lat-dist4*: $(x + y) \sqcap z = (x \sqcap z) + (y \sqcap z)$
  ⟨*proof*⟩

lemma *d-equiv-prop*: $(\forall z.\ z + x = z + y \wedge z \sqcap x = z \sqcap y) \Longrightarrow x = y$
  ⟨*proof*⟩

end

The symbol $\overline{1}_\pi$ from [2] is written nc in this theory file.

class *c-lattice* = *pbdl-monoid* +
  fixes *nc* :: $'a$
  assumes *cl1* [*simp*]: $x \cdot 1_\pi + x \cdot nc = x \cdot U$
  and *cl2* [*simp*]: $1_\pi \sqcap (x + nc) = x \cdot 0$
  and *cl3*: $x \cdot (y \parallel z) \leq (x \cdot y) \parallel (x \cdot z)$
  and *cl4*: $z \parallel z \leq z \Longrightarrow (x \parallel y) \cdot z = (x \cdot z) \parallel (y \cdot z)$

and *cl5*: $x \cdot (y \cdot (z \cdot 0)) = (x \cdot y) \cdot (z \cdot 0)$
and *cl6* [*simp*]: $(x \cdot 0) \cdot z = x \cdot 0$
and *cl7* [*simp*]: $1_\sigma \parallel 1_\sigma = 1_\sigma$
and *cl8* [*simp*]: $((x \cdot 1_\pi) \parallel 1_\sigma) \cdot y = (x \cdot 1_\pi) \parallel y$
and *cl9* [*simp*]: $((x \sqcap 1_\sigma) \cdot 1_\pi) \parallel 1_\sigma = x \sqcap 1_\sigma$
and *cl10*: $((x \sqcap nc) \cdot 1_\pi) \parallel 1_\sigma = 1_\sigma \sqcap (x \sqcap nc) \cdot nc$
and *cl11* [*simp*]: $((x \sqcap nc) \cdot 1_\pi) \parallel nc = (x \sqcap nc) \cdot nc$

begin

We show that every c-lattice is a c-trioid (Proposition 7.1) Proposition 7.2 is again covered by the theory for multirelations.

subclass *c-trioid*
⟨*proof*⟩

First we verify the complementation conditions after the definition of c-lattices.

lemma *c-nc-comp1* [*simp*]: $1_\pi + nc = U$
⟨*proof*⟩

lemma *c-nc-comp2* [*simp*]: $1_\pi \sqcap nc = 0$
⟨*proof*⟩

lemma *c-0*: $x \sqcap 1_\pi = x \cdot 0$
⟨*proof*⟩

Next we verify the conditions in Proposition 7.2.

lemma *d-s-subid*: $d\ x = x \longleftrightarrow x \leq 1_\sigma$
⟨*proof*⟩

lemma *term-p-subid*: $x \cdot 1_\pi = x \longleftrightarrow x \leq 1_\pi$
⟨*proof*⟩

lemma *term-p-subid-var*: $x \cdot 0 = x \longleftrightarrow x \leq 1_\pi$
⟨*proof*⟩

lemma *vec-iff*: $d\ x \cdot U = x \longleftrightarrow (x \cdot 1_\pi) \parallel U = x$
⟨*proof*⟩

lemma *nc-iff1*: $x \leq nc \longleftrightarrow x \sqcap 1_\pi = 0$
⟨*proof*⟩

lemma *nc-iff2*: $x \leq nc \longleftrightarrow x \cdot 0 = 0$
⟨*proof*⟩

The results of Lemma 7.3 are again at the multirelational level. Hence we continue with Lemma 7.4.

lemma *assoc-p-subid*: $(x \cdot y) \cdot (z \cdot 1_\pi) = x \cdot (y \cdot (z \cdot 1_\pi))$

⟨*proof*⟩

lemma *zero-assoc3*: $(x \cdot y) \cdot 0 = x \cdot (y \cdot 0)$
⟨*proof*⟩

lemma *x-zero-interr*: $(x \cdot 0) \parallel (y \cdot 0) = (x \parallel y) \cdot 0$
⟨*proof*⟩

lemma *p-subid-interr*: $(x \cdot z \cdot 1_\pi) \parallel (y \cdot z \cdot 1_\pi) = (x \parallel y) \cdot z \cdot 1_\pi$
⟨*proof*⟩

lemma *d-interr*: $(x \cdot d\ z) \parallel (y \cdot d\ z) = (x \parallel y) \cdot d\ z$
⟨*proof*⟩

lemma *subidem-par*: $x \leq x \parallel x$
⟨*proof*⟩

lemma *meet-le-par*: $x \sqcap y \leq x \parallel y$
⟨*proof*⟩

Next we verify Lemma 7.5 and prove some related properties.

lemma *x-split* [*simp*]: $(x \sqcap nc) + (x \sqcap 1_\pi) = x$
⟨*proof*⟩

lemma *x-split-var* [*simp*]: $(x \sqcap nc) + (x \cdot 0) = x$
⟨*proof*⟩

lemma *s-subid-closed* [*simp*]: $x \sqcap nc \sqcap 1_\sigma = x \sqcap 1_\sigma$
⟨*proof*⟩

lemma *sub-id-le-nc*: $x \sqcap 1_\sigma \leq nc$
⟨*proof*⟩

lemma *s-x-c* [*simp*]: $1_\sigma \sqcap (x \cdot 1_\pi) = 0$
⟨*proof*⟩

lemma *s-x-zero* [*simp*]: $1_\sigma \sqcap (x \cdot 0) = 0$
⟨*proof*⟩

lemma *c-nc* [*simp*]: $(x \cdot 1_\pi) \sqcap nc = 0$
⟨*proof*⟩

lemma *zero-nc* [*simp*]: $(x \cdot 0) \sqcap nc = 0$
⟨*proof*⟩

lemma *nc-zero* [*simp*]: $(x \sqcap nc) \cdot 0 = 0$
⟨*proof*⟩

Lemma 7.6.

lemma *c-def* [*simp*]: $U \cdot 0 = 1_\pi$
⟨*proof*⟩

lemma *c-x-prop* [*simp*]: $1_\pi \cdot x = 1_\pi$
⟨*proof*⟩

lemma *U-idem-s-prod* [*simp*]: $U \cdot U = U$
⟨*proof*⟩

lemma *U-idem-p-prod* [*simp*]: $U \parallel U = U$
⟨*proof*⟩

lemma *U-c* [*simp*]: $U \cdot 1_\pi = 1_\pi$
⟨*proof*⟩

lemma *s-le-nc*: $1_\sigma \leq nc$
⟨*proof*⟩

lemma *nc-c* [*simp*]: $nc \cdot 1_\pi = 1_\pi$
⟨*proof*⟩

lemma *nc-nc* [*simp*]: $nc \cdot nc = nc$
⟨*proof*⟩

lemma *U-nc* [*simp*]: $U \cdot nc = U$
⟨*proof*⟩

lemma *nc-U* [*simp*]: $nc \cdot U = U$
⟨*proof*⟩

lemma *nc-nc-par* [*simp*]: $nc \parallel nc = nc$
⟨*proof*⟩

lemma *U-nc-par* [*simp*]: $U \parallel nc = nc$
⟨*proof*⟩

We prove Lemma 7.8 and related properties.

lemma *x-y-split* [*simp*]: $(x \sqcap nc) \cdot y + x \cdot 0 = x \cdot y$
⟨*proof*⟩

lemma *x-y-prop*: $1_\sigma \sqcap (x \sqcap nc) \cdot y = 1_\sigma \sqcap x \cdot y$
⟨*proof*⟩

lemma *s-nc-U*: $1_\sigma \sqcap x \cdot nc = 1_\sigma \sqcap x \cdot U$
⟨*proof*⟩

lemma *sid-le-nc-var*: $1_\sigma \sqcap x \leq 1_\sigma \sqcap x \parallel nc$
⟨*proof*⟩

lemma *s-nc-par-U*: $1_\sigma \sqcap x \parallel nc = 1_\sigma \sqcap x \parallel U$
⟨*proof*⟩

lemma *x-c-nc-split*: $(x \cdot 1_\pi) \parallel nc = (x \sqcap nc) \cdot nc + (x \cdot 0) \parallel nc$
⟨*proof*⟩

lemma *x-c-U-split*: $(x \cdot 1_\pi) \parallel U = x \cdot U + (x \cdot 0) \parallel U$
⟨*proof*⟩

2.5 Domain in C-Lattices

We now prove variants of the domain axioms and verify the properties of Section 8 in [2].

lemma *cl9-d* [*simp*]: $d\ (x \sqcap 1_\sigma) = x \sqcap 1_\sigma$
⟨*proof*⟩

lemma *cl10-d*: $d\ (x \sqcap nc) = 1_\sigma \sqcap (x \sqcap nc) \cdot nc$
⟨*proof*⟩

lemma *cl11-d* [*simp*]: $d\ (x \sqcap nc) \cdot nc = (x \sqcap nc) \cdot nc$
⟨*proof*⟩

lemma *cl10-d-var1*: $d\ (x \sqcap nc) = 1_\sigma \sqcap x \cdot nc$
⟨*proof*⟩

lemma *cl10-d-var2*: $d\ (x \sqcap nc) = 1_\sigma \sqcap (x \sqcap nc) \cdot U$
⟨*proof*⟩

lemma *cl10-d-var3*: $d\ (x \sqcap nc) = 1_\sigma \sqcap x \cdot U$
⟨*proof*⟩

We verify the remaining properties of Lemma 8.1.

lemma *d-U* [*simp*]: $d\ U = 1_\sigma$
⟨*proof*⟩

lemma *d-nc* [*simp*]: $d\ nc = 1_\sigma$
⟨*proof*⟩

lemma *alt-d-def-nc-nc*: $d\ (x \sqcap nc) = 1_\sigma \sqcap ((x \sqcap nc) \cdot 1_\pi) \parallel nc$
⟨*proof*⟩

lemma *alt-d-def-nc-U*: $d\ (x \sqcap nc) = 1_\sigma \sqcap ((x \sqcap nc) \cdot 1_\pi) \parallel U$
⟨*proof*⟩

We verify the identity before Lemma 8.2 of [2] together with variants.

lemma *d-def-split* [*simp*]: $d\ (x \sqcap nc) + d\ (x \cdot 0) = d\ x$
⟨*proof*⟩

lemma *d-def-split-var* [*simp*]: $d\ (x \sqcap nc) + (x \cdot 0) \parallel 1_\sigma = d\ x$
⟨*proof*⟩

lemma *ax7* [*simp*]: $(1_\sigma \sqcap x \cdot U) + (x \cdot 0) \parallel 1_\sigma = d\ x$
⟨*proof*⟩

Lemma 8.2.

lemma *dom12-d*: $d\ x = 1_\sigma \sqcap (x \cdot 1_\pi) \parallel nc$
⟨*proof*⟩

lemma *dom12-d-U*: $d\ x = 1_\sigma \sqcap (x \cdot 1_\pi) \parallel U$
⟨*proof*⟩

lemma *dom-def-var*: $d\ x = (x \cdot U \sqcap 1_\pi) \parallel 1_\sigma$
⟨*proof*⟩

Lemma 8.3.

lemma *ax5-d* [*simp*]: $d\ (x \sqcap nc) \cdot U = (x \sqcap nc) \cdot U$
⟨*proof*⟩

lemma *ax5-0* [*simp*]: $d\ (x \cdot 0) \cdot U = (x \cdot 0) \parallel U$
⟨*proof*⟩

lemma *x-c-U-split2*: $d\ x \cdot nc = (x \sqcap nc) \cdot nc + (x \cdot 0) \parallel nc$
⟨*proof*⟩

lemma *x-c-U-split3*: $d\ x \cdot U = (x \sqcap nc) \cdot U + (x \cdot 0) \parallel U$
⟨*proof*⟩

lemma *x-c-U-split-d*: $d\ x \cdot U = x \cdot U + (x \cdot 0) \parallel U$
⟨*proof*⟩

lemma *x-U-prop2*: $x \cdot nc = d\ (x \sqcap nc) \cdot nc + x \cdot 0$
⟨*proof*⟩

lemma *x-U-prop3*: $x \cdot U = d\ (x \sqcap nc) \cdot U + x \cdot 0$
⟨*proof*⟩

lemma *d-x-nc* [*simp*]: $d\ (x \cdot nc) = d\ x$
⟨*proof*⟩

lemma *d-x-U* [*simp*]: $d\ (x \cdot U) = d\ x$
⟨*proof*⟩

The next properties of domain are important, but do not feature in [2]. Proofs can be found in [1].

lemma *d-llp1*: $d\ x \leq d\ y \Longrightarrow x \leq d\ y \cdot x$
⟨*proof*⟩

lemma *d-llp2*: $x \leq d\ y \cdot x \Longrightarrow d\ x \leq d\ y$
⟨*proof*⟩

lemma *demod1*: $d\ (x \cdot y) \leq d\ z \Longrightarrow x \cdot d\ y \leq d\ z \cdot x$
⟨*proof*⟩

lemma *demod2*: $x \cdot d\ y \leq d\ z \cdot x \Longrightarrow d\ (x \cdot y) \leq d\ z$
⟨*proof*⟩

2.6 Structural Properties of C-Lattices

Now we consider the results from Section 9 and 10 in [2]. First we verify the conditions for Proposition 9.1.

lemma *d-meet-closed* [*simp*]: $d\ (d\ x \sqcap d\ y) = d\ x \sqcap d\ y$
⟨*proof*⟩

lemma *d-s-prod-eq-meet*: $d\ x \cdot d\ y = d\ x \sqcap d\ y$
⟨*proof*⟩

lemma *d-p-prod-eq-meet*: $d\ x \parallel d\ y = d\ x \sqcap d\ y$
⟨*proof*⟩

lemma *s-id-par-s-prod*: $(x \sqcap 1_\sigma) \parallel (y \sqcap 1_\sigma) = (x \sqcap 1_\sigma) \cdot (y \sqcap 1_\sigma)$
⟨*proof*⟩

lemma *s-id-par* [*simp*]: $x \sqcap 1_\sigma \parallel x \sqcap 1_\sigma = x \sqcap 1_\sigma$
⟨*proof*⟩

We verify the remaining conditions in Proposition 9.2.

lemma *p-subid-par-eq-meet*: $(x \cdot 0) \parallel (y \cdot 0) = (x \cdot 0) \sqcap (y \cdot 0)$
⟨*proof*⟩

lemma *p-subid-par-eq-meet-var*: $(x \cdot 1_\pi) \parallel (y \cdot 1_\pi) = (x \cdot 1_\pi) \sqcap (y \cdot 1_\pi)$
⟨*proof*⟩

lemma *x-zero-add-closed*: $x \cdot 0 + y \cdot 0 = (x + y) \cdot 0$
⟨*proof*⟩

lemma *x-zero-meet-closed*: $(x \cdot 0) \sqcap (y \cdot 0) = (x \sqcap y) \cdot 0$
⟨*proof*⟩

The following set of lemmas investigates the closure properties of vectors, including Lemma 9,3.

lemma *U-par-zero* [*simp*]: $(0 \cdot c) \parallel U = 0$
⟨*proof*⟩

lemma *U-par-s-id* [*simp*]: $(1_\sigma \cdot 1_\pi) \parallel U = U$
⟨*proof*⟩

lemma *U-par-p-id* [*simp*]: $(1_\pi \cdot 1_\pi) \parallel U = U$
⟨*proof*⟩

lemma *U-par-nc* [*simp*]: $(nc \cdot 1_\pi) \parallel U = U$
⟨*proof*⟩

lemma *d-add-var*: $d\ x \cdot z + d\ y \cdot z = d\ (x + y) \cdot z$
⟨*proof*⟩

lemma *d-interr-U*: $(d\ x \cdot U) \parallel (d\ y \cdot U) = d\ (x \parallel y) \cdot U$
⟨*proof*⟩

lemma *d-meet*:
assumes $\bigwedge x\ y\ z.\ (x \sqcap y \sqcap 1_\sigma) \cdot z = (x \sqcap 1_\sigma) \cdot z \sqcap (y \sqcap 1_\sigma) \cdot z$
shows $d\ x \cdot z \sqcap d\ y \cdot z = (d\ x \sqcap d\ y) \cdot z$
⟨*proof*⟩

Proposition 9.4

lemma *nc-zero-closed* [*simp*]: $0 \sqcap nc = 0$
⟨*proof*⟩

lemma *nc-s* [*simp*]: $1_\sigma \sqcap nc = 1_\sigma$
⟨*proof*⟩

lemma *nc-add-closed*: $(x \sqcap nc) + (y \sqcap nc) = (x + y) \sqcap nc$
⟨*proof*⟩

lemma *nc-meet-closed*: $(x \sqcap nc) \sqcap (y \sqcap nc) = x \sqcap y \sqcap nc$
⟨*proof*⟩

lemma *nc-scomp-closed*: $((x \sqcap nc) \cdot (y \sqcap nc)) \leq nc$
⟨*proof*⟩

lemma *nc-scomp-closed-alt* [*simp*]: $((x \sqcap nc) \cdot (y \sqcap nc)) \sqcap nc = (x \sqcap nc) \cdot (y \sqcap nc)$
⟨*proof*⟩

lemma *nc-ccomp-closed*: $(x \sqcap nc) \parallel (y \sqcap nc) \leq nc$
⟨*proof*⟩

lemma *nc-ccomp-closed-alt* [*simp*]: $(x \parallel (y \sqcap nc)) \sqcap nc = x \parallel (y \sqcap nc)$
⟨*proof*⟩

Lemma 9.6.

lemma *tarski-prod*:
assumes $\bigwedge x.\ x \sqcap nc \neq 0 \implies nc \cdot ((x \sqcap nc) \cdot nc) = nc$
and $\bigwedge x\ y\ z.\ d\ x \cdot (y \cdot z) = (d\ x \cdot y) \cdot z$
shows $((x \sqcap nc) \cdot nc) \cdot ((y \sqcap nc) \cdot nc) = (\mathit{if}\ \ (y \sqcap nc)\ \ = 0\ \mathit{then}\ 0\ \mathit{else}\ (x \sqcap nc)$

$\cdot$ *nc*)
⟨*proof*⟩

We show the remaining conditions of Proposition 9.8.

lemma *nc-prod-aux* [*simp*]: $((x \sqcap nc) \cdot nc) \cdot nc = (x \sqcap nc) \cdot nc$
⟨*proof*⟩

lemma *nc-vec-add-closed*: $((x \sqcap nc) \cdot nc + (y \sqcap nc) \cdot nc) \cdot nc = (x \sqcap nc) \cdot nc + (y \sqcap nc) \cdot nc$
⟨*proof*⟩

lemma *nc-vec-par-closed*: $(((x \sqcap nc) \cdot nc) \parallel ((y \sqcap nc) \cdot nc)) \cdot nc = ((x \sqcap nc) \cdot nc) \parallel ((y \sqcap nc) \cdot nc)$
⟨*proof*⟩

lemma *nc-vec-par-is-meet*:
assumes $\bigwedge x\ y\ z.\ (d\ x \sqcap d\ y) \cdot z = d\ x \cdot z \sqcap d\ y \cdot z$
shows $((x \sqcap nc) \cdot nc) \parallel ((y \sqcap nc) \cdot nc) = ((x \sqcap nc) \cdot nc) \sqcap ((y \sqcap nc) \cdot nc)$
⟨*proof*⟩

lemma *nc-vec-meet-closed*:
assumes $\bigwedge x\ y\ z.\ (d\ x \sqcap d\ y) \cdot z = d\ x \cdot z \sqcap d\ y \cdot z$
shows $((x \sqcap nc) \cdot nc \sqcap (y \sqcap nc) \cdot nc) \cdot nc = (x \sqcap nc) \cdot nc \sqcap (y \sqcap nc) \cdot nc$
⟨*proof*⟩

lemma *nc-vec-seq-closed*:
assumes $\bigwedge x.\ x \sqcap nc \neq 0 \Longrightarrow nc \cdot ((x \sqcap nc) \cdot nc) = nc$
and $\bigwedge x\ y\ z.\ d\ x \cdot (y \cdot z) = (d\ x \cdot y) \cdot z$
shows $(((x \sqcap nc) \cdot nc) \cdot ((y \sqcap nc) \cdot nc)) \cdot nc = ((x \sqcap nc) \cdot nc) \cdot ((y \sqcap nc) \cdot nc)$
⟨*proof*⟩

Proposition 10.1 and 10.2.

lemma *iso3* [*simp*]: $d\ (d\ x \cdot U) = d\ x$
⟨*proof*⟩

lemma *iso4* [*simp*]: $d\ ((x \cdot 1_\pi) \parallel U) \cdot U = (x \cdot 1_\pi) \parallel U$
⟨*proof*⟩

lemma *iso5* [*simp*]: $((x \cdot 1_\pi) \parallel U) \cdot 1_\pi = x \cdot 1_\pi$
⟨*proof*⟩

lemma *iso6* [*simp*]: $(((x \cdot 1_\pi) \parallel U) \cdot 1_\pi) \parallel U = (x \cdot 1_\pi) \parallel U$
⟨*proof*⟩

lemma *iso3-sharp* [*simp*]: $d\ (d\ (x \sqcap nc) \cdot nc) = d\ (x \sqcap nc)$
⟨*proof*⟩

lemma *iso4-sharp* [*simp*]: $d\ ((x \sqcap nc) \cdot nc) \cdot nc = (x \sqcap nc) \cdot nc$
⟨*proof*⟩

lemma *iso5-sharp* [*simp*]: $(((x \sqcap nc) \cdot 1_\pi) \parallel nc) \cdot 1_\pi = (x \sqcap nc) \cdot 1_\pi$
⟨*proof*⟩

lemma *iso6-sharp* [*simp*]: $(((x \sqcap nc) \cdot nc) \cdot 1_\pi) \parallel nc = (x \sqcap nc) \cdot nc$
⟨*proof*⟩

We verify Lemma 15.2 at this point, because it is helpful for the following proofs.

lemma *uc-par-meet*: $x \parallel U \sqcap y \parallel U = x \parallel U \parallel y \parallel U$
⟨*proof*⟩

lemma *uc-unc* [*simp*]: $x \parallel U \parallel x \parallel U = x \parallel U$
⟨*proof*⟩

lemma *uc-interr*: $(x \parallel y) \cdot (z \parallel U) = (x \cdot (z \parallel U)) \parallel (y \cdot (z \parallel U))$
⟨*proof*⟩

We verify the remaining cases of Proposition 10.3.

lemma *sc-hom-meet*: $(d\ x \sqcap d\ y) \cdot 1_\pi = (d\ x) \cdot 1_\pi \sqcap (d\ y) \cdot 1_\pi$
⟨*proof*⟩

lemma *sc-hom-seq*: $(d\ x \cdot d\ y) \cdot 1_\pi = (d\ x \sqcap d\ y) \cdot 1_\pi$
⟨*proof*⟩

lemma *cs-hom-meet*: $d\ (x \cdot 1_\pi \sqcap y \cdot 1_\pi) = d\ (x \cdot 1_\pi) \sqcap d\ (y \cdot 1_\pi)$
⟨*proof*⟩

lemma *sv-hom-meet*: $(d\ x \sqcap d\ y) \cdot U = (d\ x) \cdot U \sqcap (d\ y) \cdot U$
⟨*proof*⟩

lemma *sv-hom-par*: $(x \parallel y) \cdot U = (x \cdot U) \parallel (y \cdot U)$
⟨*proof*⟩

lemma *vs-hom-meet*: $d\ (((x \cdot 1_\pi) \parallel U) \sqcap ((y \cdot 1_\pi) \parallel U)) = d\ ((x \cdot 1_\pi) \parallel U) \sqcap d\ ((y \cdot 1_\pi) \parallel U)$
⟨*proof*⟩

lemma *cv-hom-meet*: $(x \cdot 1_\pi \sqcap y \cdot 1_\pi) \parallel U = (x \cdot 1_\pi) \parallel U \sqcap (y \cdot 1_\pi) \parallel U$
⟨*proof*⟩

lemma *cv-hom-par* [*simp*]: $x \parallel U \parallel y \parallel U = (x \parallel y) \parallel U$
⟨*proof*⟩

lemma *vc-hom-meet*: $((x \cdot 1_\pi) \parallel U \sqcap (y \cdot 1_\pi) \parallel U) \cdot 1_\pi = ((x \cdot 1_\pi) \parallel U) \cdot 1_\pi \sqcap ((y \cdot 1_\pi) \parallel U) \cdot 1_\pi$
⟨*proof*⟩

lemma *vc-hom-seq*: $(((x \cdot 1_\pi) \parallel U) \cdot ((y \cdot 1_\pi) \parallel U)) \cdot 1_\pi = (((x \cdot 1_\pi) \parallel U) \cdot 1_\pi) \cdot (((y \cdot 1_\pi) \parallel U) \cdot 1_\pi)$
⟨*proof*⟩

Proposition 10.4.

lemma *nsv-hom-meet*: $(d\ x \sqcap d\ y) \cdot nc = (d\ x) \cdot nc \sqcap (d\ y) \cdot nc$
⟨*proof*⟩

lemma *nsv-hom-par*: $(x \parallel y) \cdot nc = (x \cdot nc) \parallel (y \cdot nc)$
⟨*proof*⟩

lemma *vec-p-prod-meet*: $((x \sqcap nc) \cdot nc) \parallel ((y \sqcap nc) \cdot nc) = ((x \sqcap nc) \cdot\ nc) \sqcap ((y \sqcap nc) \cdot nc)$
⟨*proof*⟩

lemma *nvs-hom-meet*: $d\ (((x \sqcap nc) \cdot nc) \sqcap ((y \sqcap nc) \cdot nc)) = d\ ((x \sqcap nc) \cdot nc) \sqcap d\ ((y \sqcap nc) \cdot nc)$
⟨*proof*⟩

lemma *ncv-hom-meet*: $(x \cdot 1_\pi \sqcap y \cdot 1_\pi) \parallel nc = (x \cdot 1_\pi) \parallel nc \sqcap (y \cdot 1_\pi) \parallel nc$
⟨*proof*⟩

lemma *ncv-hom-par*: $(x \parallel y) \parallel nc = x \parallel nc \parallel y \parallel nc$
⟨*proof*⟩

lemma *nvc-hom-meet*: $((x \sqcap nc) \cdot\ nc \sqcap (y \sqcap nc) \cdot nc) \cdot 1_\pi = ((x \sqcap nc) \cdot\ nc) \cdot 1_\pi \sqcap ((y \sqcap nc) \cdot nc) \cdot 1_\pi$
⟨*proof*⟩

2.7 Terminal and Nonterminal Elements

Now we define the projection functions on terminals and nonterminal parts and verify the properties of Section 11 in [2].

definition *tau* :: $'a \Rightarrow 'a$ (τ) **where**
$\tau\ x = x \cdot 0$

definition *nu* :: $'a \Rightarrow 'a$ (ν) **where**
$\nu\ x = x \sqcap nc$

Lemma 11.1.

lemma *tau-int*: $\tau\ x \leq x$
⟨*proof*⟩

lemma *nu-int*: $\nu\ x \leq x$
⟨*proof*⟩

lemma *tau-ret* [*simp*]: $\tau\ (\tau\ x) = \tau\ x$
⟨*proof*⟩

lemma *nu-ret* [*simp*]: $\nu\ (\nu\ x) = \nu\ x$
⟨*proof*⟩

lemma *tau-iso*: $x \leq y \Longrightarrow \tau\ x \leq \tau\ y$
⟨*proof*⟩

lemma *nu-iso*: $x \leq y \Longrightarrow \nu\ x \leq \nu\ y$
⟨*proof*⟩

Lemma 11.2.

lemma *tau-zero* [*simp*]: $\tau\ 0 = 0$
⟨*proof*⟩

lemma *nu-zero* [*simp*]: $\nu\ 0 = 0$
⟨*proof*⟩

lemma *tau-s* [*simp*]: $\tau\ 1_\sigma = 0$
⟨*proof*⟩

lemma *nu-s* [*simp*]: $\nu\ 1_\sigma = 1_\sigma$
⟨*proof*⟩

lemma *tau-c* [*simp*]: $\tau\ 1_\pi = 1_\pi$
⟨*proof*⟩

lemma *nu-c* [*simp*]: $\nu\ 1_\pi = 0$
⟨*proof*⟩

lemma *tau-nc* [*simp*]: $\tau\ nc = 0$
⟨*proof*⟩

lemma *nu-nc* [*simp*]: $\nu\ nc = nc$
⟨*proof*⟩

lemma *tau-U* [*simp*]: $\tau\ U = 1_\pi$
⟨*proof*⟩

lemma *nu-U* [*simp*]: $\nu\ U = nc$
⟨*proof*⟩

Lemma 11.3.

lemma *tau-add* [*simp*]: $\tau\ (x + y) = \tau\ x + \tau\ y$
⟨*proof*⟩

lemma *nu-add* [*simp*]: $\nu\ (x + y) = \nu\ x + \nu\ y$
⟨*proof*⟩

lemma *tau-meet* [*simp*]: $\tau\ (x \sqcap y) = \tau\ x \sqcap \tau\ y$

⟨*proof*⟩

lemma *nu-meet* [*simp*]: $\nu\ (x \sqcap y) = \nu\ x \sqcap \nu\ y$
⟨*proof*⟩

lemma *tau-seq*: $\tau\ (x \cdot y) = \tau\ x + \nu\ x \cdot \tau\ y$
⟨*proof*⟩

lemma *tau-par* [*simp*]: $\tau\ (x \parallel y) = \tau\ x \parallel \tau\ y$
⟨*proof*⟩

lemma *nu-par-aux1*: $x \parallel \tau\ y = d\ (\tau\ y) \cdot x$
⟨*proof*⟩

lemma *nu-par-aux2* [*simp*]: $\nu\ (\nu\ x \parallel \nu\ y) = \nu\ x \parallel \nu\ y$
⟨*proof*⟩

lemma *nu-par-aux3* [*simp*]: $\nu\ (\nu\ x \parallel \tau\ y) = \nu\ x \parallel \tau\ y$
⟨*proof*⟩

lemma *nu-par-aux4* [*simp*]: $\nu\ (\tau\ x \parallel \tau\ y) = 0$
⟨*proof*⟩

lemma *nu-par*: $\nu\ (x \parallel y) = d\ (\tau\ x) \cdot \nu\ y + d\ (\tau\ y) \cdot \nu\ x + \nu\ x \parallel \nu\ y$
⟨*proof*⟩

Lemma 11.5.

lemma *sprod-tau-nu*: $x \cdot y = \tau\ x + \nu\ x \cdot y$
⟨*proof*⟩

lemma *pprod-tau-nu*: $x \parallel y = \nu\ x \parallel \nu\ y + d\ (\tau\ x) \cdot \nu\ y + d\ (\tau\ y) \cdot \nu\ x + \tau\ x \parallel \tau\ y$
⟨*proof*⟩

We now verify some additional properties which are not mentioned in the paper.

lemma *tau-idem* [*simp*]: $\tau\ x \cdot \tau\ x = \tau\ x$
⟨*proof*⟩

lemma *tau-interr*: $(x \parallel y) \cdot \tau\ z = (x \cdot \tau\ z) \parallel (y \cdot \tau\ z)$
⟨*proof*⟩

lemma *tau-le-c*: $\tau\ x \leq 1_\pi$
⟨*proof*⟩

lemma *c-le-tauc*: $1_\pi \leq \tau\ 1_\pi$
⟨*proof*⟩

lemma *x-alpha-tau* [*simp*]: $\nu\ x + \tau\ x = x$

⟨*proof*⟩

lemma *alpha-tau-zero* [*simp*]: $\nu\ (\tau\ x) = 0$
⟨*proof*⟩

lemma *tau-alpha-zero* [*simp*]: $\tau\ (\nu\ x) = 0$
⟨*proof*⟩

lemma *sprod-tau-nu-var* [*simp*]: $\nu\ (\nu\ x \cdot y) = \nu\ (x \cdot y)$
⟨*proof*⟩

lemma *tau-s-prod* [*simp*]: $\tau\ (x \cdot y) = x \cdot \tau\ y$
⟨*proof*⟩

lemma *alpha-fp*: $\nu\ x = x \longleftrightarrow x \cdot 0 = 0$
⟨*proof*⟩

lemma *alpha-prod-closed* [*simp*]: $\nu\ (\nu\ x \cdot \nu\ y) = \nu\ x \cdot \nu\ y$
⟨*proof*⟩

lemma *alpha-par-prod* [*simp*]: $\nu\ (x \parallel \nu\ y) = x \parallel \nu\ y$
⟨*proof*⟩

lemma *p-prod-tau-alpha*: $x \parallel y = x \parallel \nu\ y + \nu\ x \parallel y + \tau\ x \parallel \tau\ y$
⟨*proof*⟩

lemma *p-prod-tau-alpha-var*: $x \parallel y = x \parallel \nu\ y + \nu\ x \parallel y + \tau\ (x \parallel y)$
⟨*proof*⟩

lemma *alpha-par*: $\nu\ (x \parallel y) = \nu\ x \parallel y + x \parallel \nu\ y$
⟨*proof*⟩

lemma *alpha-tau* [*simp*]: $\nu\ (x \cdot \tau\ y) = 0$
⟨*proof*⟩

lemma *nu-par-prop*: $\nu\ x = x \Longrightarrow \nu\ (x \parallel y) = x \parallel y$
⟨*proof*⟩

lemma *tau-seq-prop*: $\tau\ x = x \Longrightarrow x \cdot y\ = x$
⟨*proof*⟩

lemma *tau-seq-prop2*: $\tau\ y = y \Longrightarrow \tau\ (x \cdot y)\ = x \cdot y$
⟨*proof*⟩

lemma *d-nu*: $\nu\ (d\ x \cdot y) = d\ x \cdot \nu\ y$
⟨*proof*⟩

Lemma 11.6 and 11.7.

lemma *nu-ideal1*: $[\![\nu\ x = x;\ y \leq x]\!] \Longrightarrow \nu\ y = y$

⟨*proof*⟩

lemma *tau-ideal1*: $[\![\tau\ x = x;\ y \leq x]\!] \Longrightarrow \tau\ y = y$
⟨*proof*⟩

lemma *nu-ideal2*: $[\![\nu\ x = x;\ \nu\ y = y]\!] \Longrightarrow \nu\ (x + y) = x + y$
⟨*proof*⟩

lemma *tau-ideal2*: $[\![\tau\ x = x;\ \tau\ y = y]\!] \Longrightarrow \tau\ (x + y) = x + y$
⟨*proof*⟩

lemma *tau-ideal3*: $\tau\ x = x \Longrightarrow \tau\ (x \cdot y) = x \cdot y$
⟨*proof*⟩

We prove the precongruence properties of Lemma 11.9.

lemma *tau-add-precong*: $\tau\ x \leq \tau\ y \Longrightarrow \tau\ (x + z) \leq \tau\ (y + z)$
⟨*proof*⟩

lemma *tau-meet-precong*: $\tau\ x \leq \tau\ y \Longrightarrow \tau\ (x \sqcap z) \leq \tau\ (y \sqcap z)$
⟨*proof*⟩

lemma *tau-par-precong*: $\tau\ x \leq \tau\ y \Longrightarrow \tau\ (x \parallel z) \leq \tau\ (y \parallel z)$
⟨*proof*⟩

lemma *tau-seq-precongl*: $\tau\ x \leq \tau\ y \Longrightarrow \tau\ (z \cdot x) \leq \tau\ (z \cdot y)$
⟨*proof*⟩

lemma *nu-add-precong*: $\nu\ x \leq \nu\ y \Longrightarrow \nu\ (x + z) \leq \nu\ (y + z)$
⟨*proof*⟩

lemma *nu-meet-precong*: $\nu\ x \leq \nu\ y \Longrightarrow \nu\ (x \sqcap z) \leq \nu\ (y \sqcap z)$
⟨*proof*⟩

lemma *nu-seq-precongr*: $\nu\ x \leq \nu\ y \Longrightarrow \nu\ (x \cdot z) \leq \nu\ (y \cdot z)$
⟨*proof*⟩

We prove the congruence properties of Corollary 11.11.

definition *tcg* :: $'a \Rightarrow 'a \Rightarrow bool$ **where**
$tcg\ x\ y = (\tau\ x \leq \tau\ y \wedge \tau\ y \leq \tau\ x)$

definition *ncg* :: $'a \Rightarrow 'a \Rightarrow bool$ **where**
$ncg\ x\ y = (\nu\ x \leq \nu\ y \wedge \nu\ y \leq \nu\ x)$

lemma *tcg-refl*: $tcg\ x\ x$
⟨*proof*⟩

lemma *tcg-trans*: $[\![tcg\ x\ y;\ tcg\ y\ z]\!] \Longrightarrow tcg\ x\ z$
⟨*proof*⟩

lemma *tcg-sym*: $tcg\ x\ y \Longrightarrow tcg\ y\ x$
⟨*proof*⟩

lemma *ncg-refl*: $ncg\ x\ x$
⟨*proof*⟩

lemma *ncg-trans*: $[\![ncg\ x\ y;\ ncg\ y\ z]\!] \Longrightarrow ncg\ x\ z$
⟨*proof*⟩

lemma *ncg-sym*: $ncg\ x\ y \Longrightarrow ncg\ y\ x$
⟨*proof*⟩

lemma *tcg-alt*: $tcg\ x\ y = (\tau\ x = \tau\ y)$
⟨*proof*⟩

lemma *ncg-alt*: $ncg\ x\ y = (\nu\ x = \nu\ y)$
⟨*proof*⟩

lemma *tcg-add*: $\tau\ x = \tau\ y \Longrightarrow \tau\ (x + z) = \tau\ (y + z)$
⟨*proof*⟩

lemma *tcg-meet*: $\tau\ x = \tau\ y \Longrightarrow \tau\ (x \sqcap z) = \tau\ (y \sqcap z)$
⟨*proof*⟩

lemma *tcg-par*: $\tau\ x = \tau\ y \Longrightarrow \tau\ (x \parallel z) = \tau\ (y \parallel z)$
⟨*proof*⟩

lemma *tcg-seql*: $\tau\ x = \tau\ y \Longrightarrow \tau\ (z \cdot x) = \tau\ (z \cdot y)$
⟨*proof*⟩

lemma *ncg-add*: $\nu\ x = \nu\ y \Longrightarrow \nu\ (x + z) = \nu\ (y + z)$
⟨*proof*⟩

lemma *ncg-meet*: $\nu\ x = \nu\ y \Longrightarrow \nu\ (x \sqcap z) = \nu\ (y \sqcap z)$
⟨*proof*⟩

lemma *ncg-seqr*: $\nu\ x = \nu\ y \Longrightarrow \nu\ (x \cdot z) = \nu\ (y \cdot z)$
⟨*proof*⟩

end

2.8 Powers in C-Algebras

We define the power functions from Section 6 in [2] after Lemma 12.4.

context *proto-dioid*
begin

primrec *p-power* :: $'a \Rightarrow nat \Rightarrow 'a$ **where**
p-power x 0 $= 1_\sigma$ |

p-power x (Suc n) = x · p-power x n

primrec *power-rd* :: *'a ⇒ nat ⇒ 'a* **where**
power-rd x 0 = 0 |
power-rd x (Suc n) = 1$_\sigma$ *+ x · power-rd x n*

primrec *power-sq* :: *'a ⇒ nat ⇒ 'a* **where**
power-sq x 0 = 1$_\sigma$ |
power-sq x (Suc n) = 1$_\sigma$ *+ x · power-sq x n*

Lemma 12.5

lemma *power-rd-chain*: *power-rd x n ≤ power-rd x (n + 1)*
⟨*proof*⟩

lemma *power-sq-chain*: *power-sq x n ≤ power-sq x (n + 1)*
⟨*proof*⟩

lemma *pow-chain*: *p-power (1*$_\sigma$ *+ x) n ≤ p-power (1*$_\sigma$ *+ x) (n + 1)*
⟨*proof*⟩

lemma *pow-prop*: *p-power (1*$_\sigma$ *+ x) (n + 1) = 1*$_\sigma$ *+ x · p-power (1*$_\sigma$ *+ x) n*
⟨*proof*⟩

Next we verify facts from the proofs of Lemma 12.6.

lemma *power-rd-le-sq*: *power-rd x n ≤ power-sq x n*
⟨*proof*⟩

lemma *power-sq-le-rd*: *power-sq x n ≤ power-rd x (Suc n)*
⟨*proof*⟩

lemma *power-sq-power*: *power-sq x n = p-power (1*$_\sigma$ *+ x) n*
⟨*proof*⟩

end

2.9 C-Kleene Algebras

The definition of c-Kleene algebra is slightly different from that in Section 6 of [2]. It is used to prove properties from Section 6 and Section 12.

class *c-kleene-algebra = c-lattice + star-op +*
assumes *star-unfold*: *1*$_\sigma$ *+ x · x** *≤ x**
and *star-induct*: *1*$_\sigma$ *+ x · y ≤ y ⟹ x** *≤ y*

begin

lemma *star-irr*: *1*$_\sigma$ *≤ x**
⟨*proof*⟩

lemma *star-unfold-part*: $x \cdot x^\star \leq x^\star$
⟨*proof*⟩

lemma *star-ext-aux*: $x \leq x \cdot x^\star$
⟨*proof*⟩

lemma *star-ext*: $x \leq x^\star$
⟨*proof*⟩

lemma *star-co-trans*: $x^\star \leq x^\star \cdot x^\star$
⟨*proof*⟩

lemma *star-iso*: $x \leq y \Longrightarrow x^\star \leq y^\star$
⟨*proof*⟩

lemma *star-unfold-eq* [*simp*]: $1_\sigma + x \cdot x^\star = x^\star$
⟨*proof*⟩

Lemma 12.2.

lemma *nu-star1*:
assumes $\bigwedge x\ y\ z.\ x \cdot (y \cdot z) = (x \cdot y) \cdot z$
shows $x^\star \leq (\nu\ x)^\star \cdot (1_\sigma + \tau\ x)$
⟨*proof*⟩

lemma *nu-star2*:
assumes $\bigwedge x.\ x^\star \cdot x^\star \leq x^\star$
shows $(\nu\ x)^\star \cdot (1_\sigma + \tau\ x) \leq x^\star$
⟨*proof*⟩

lemma *nu-star*:
assumes $\bigwedge x.\ x^\star \cdot x^\star \leq x^\star$
and $\bigwedge x\ y\ z.\ x \cdot (y \cdot z) = (x \cdot y) \cdot z$
shows $(\nu\ x)^\star \cdot (1_\sigma + \tau\ x) = x^\star$
⟨*proof*⟩

Lemma 12.3.

lemma *tau-star*: $(\tau\ x)^\star = 1_\sigma + \tau\ x$
⟨*proof*⟩

lemma *tau-star-var*:
assumes $\bigwedge x\ y\ z.\ x \cdot (y \cdot z) = (x \cdot y) \cdot z$
and $\bigwedge x.\ x^\star \cdot x^\star \leq x^\star$
shows $\tau\ (x^\star) = (\nu\ x)^\star \cdot \tau\ x$
⟨*proof*⟩

lemma *nu-star-sub*: $(\nu\ x)^\star \leq \nu\ (x^\star)$
⟨*proof*⟩

lemma *nu-star-nu* [*simp*]: $\nu\ ((\nu\ x)^\star) = (\nu\ x)^\star$

⟨*proof*⟩

lemma *nu-star-tau* [*simp*]: $\nu\ ((\tau\ x)^\star) = 1_\sigma$
⟨*proof*⟩

lemma *tau-star-tau* [*simp*]: $\tau\ ((\tau\ x)^\star) = \tau\ x$
⟨*proof*⟩

lemma *tau-star-nu* [*simp*]: $\tau\ ((\nu\ x)^\star) = 0$
⟨*proof*⟩

Finally we verify Lemma 6.2. Proofs can be found in [1].

lemma *d-star-unfold* [*simp*]:
assumes $\bigwedge x\ y\ z.\ (x \cdot y) \cdot d\ z = x \cdot (y \cdot d\ z)$
shows $d\ y + d\ (x \cdot d\ (x^\star \cdot y)) = d\ (x^\star \cdot y)$
⟨*proof*⟩

lemma *d-star-sim1*:
assumes $\bigwedge\ x\ y\ z.\ d\ z + x \cdot y \leq y \Longrightarrow x^\star \cdot d\ z \leq y$
and $\bigwedge\ x\ y\ z.\ (x \cdot d\ y) \cdot z = x \cdot (d\ y \cdot z)$
and $\bigwedge\ x\ y\ z.\ (d\ x \cdot y) \cdot z = d\ x \cdot (y \cdot z)$
shows $x \cdot d\ z \leq d\ z \cdot y \Longrightarrow x^\star \cdot d\ z \leq d\ z \cdot y^\star$
⟨*proof*⟩

lemma *d-star-induct*:
assumes $\bigwedge\ x\ y\ z.\ d\ z + x \cdot y \leq y \Longrightarrow x^\star \cdot d\ z \leq y$
and $\bigwedge\ x\ y\ z.\ (x \cdot d\ y) \cdot z = x \cdot (d\ y \cdot z)$
and $\bigwedge\ x\ y\ z.\ (d\ x \cdot y) \cdot z = d\ x \cdot (y \cdot z)$
shows $d\ (x \cdot y) \leq d\ y \Longrightarrow d\ (x^\star \cdot y) \leq d\ y$
⟨*proof*⟩

end

2.10 C-Omega Algebras

These structures do not feature in [2], but in fact, many lemmas from Section 13 can be proved in this setting. The proto-quantales and c-quantales using in [2] provide a more expressive setting in which least and greatest fixpoints need not be postulated; they exists due to properties of sequential composition and addition over complete lattices.

class *c-omega-algebra* = *c-kleene-algebra* + *omega-op* +
assumes *om-unfold*: $x^\omega \leq x \cdot x^\omega$
and *om-coinduct*: $y \leq x \cdot y \Longrightarrow y \leq x^\omega$

begin

Lemma 13.4.

lemma *om-unfold-eq* [*simp*]: $x \cdot x^\omega = x^\omega$

⟨*proof*⟩

lemma *om-iso*: $x \leq y \Longrightarrow x^\omega \leq y^\omega$
⟨*proof*⟩

Lemma 13.5.

lemma *zero-om* [*simp*]: $0^\omega = 0$
⟨*proof*⟩

lemma *s-id-om* [*simp*]: ${1_\sigma}^\omega = U$
⟨*proof*⟩

lemma *p-id-om* [*simp*]: ${1_\pi}^\omega = 1_\pi$
⟨*proof*⟩

lemma *nc-om* [*simp*]: $nc^\omega = U$
⟨*proof*⟩

lemma *U-om* [*simp*]: $U^\omega = U$
⟨*proof*⟩

Lemma 13.6.

lemma *tau-om1*: $\tau\ x \leq \tau\ (x^\omega)$
⟨*proof*⟩

lemma *tau-om2* [*simp*]: $\tau\ x^\omega = \tau\ x$
⟨*proof*⟩

lemma *tau-om3*: $(\tau\ x)^\omega \leq \tau\ (x^\omega)$
⟨*proof*⟩

Lemma 13.7.

lemma *om-nu-tau*: $(\nu\ x)^\omega + (\nu\ x)^\star \cdot \tau\ x \leq x^\omega$
⟨*proof*⟩

end

2.11 C-Nabla Algebras

Nabla-algebras provide yet another way of formalising non-terminating behaviour in Section 13.

class *c-nabla-algebra* = *c-omega-algebra* +
fixes *nabla* :: $'a \Rightarrow 'a$ (∇)
assumes *nabla-unfold*: $\nabla\ \ x \leq d\ (x \cdot \nabla\ x)$
and *nabla-coinduct*: $d\ y \leq d\ (x \cdot y) \Longrightarrow d\ y \leq \nabla\ x$

begin

lemma *nabla-unfold-eq* [*simp*]: $\nabla\ x = d\ (x \cdot \nabla\ x)$
⟨*proof*⟩

lemma *nabla-le-s*: $\nabla\ x \leq 1_\sigma$
⟨*proof*⟩

lemma *nabla-nu* [*simp*]: $\nu\ (\nabla\ x) = \nabla\ x$
⟨*proof*⟩

Proposition 13.9.

lemma *nabla-omega-U*:
assumes $\bigwedge x\ y\ z.\ x \cdot (d\ y \cdot z) = (x \cdot d\ y\ ) \cdot z$
shows $(\nu\ x)^\omega = \nabla\ (\nu\ x) \cdot U$
⟨*proof*⟩

Corollary 13.10.

lemma *nabla-omega-U-cor*:
assumes $\bigwedge x\ y\ z.\ x \cdot (d\ y \cdot z) = (x \cdot d\ y\ ) \cdot z$
shows $\nabla\ (\nu\ x) \cdot U + (\nu\ x)^\star \cdot \tau\ x \leq x^\omega$
⟨*proof*⟩

Lemma 13.11.

lemma *nu-om-nu*:
assumes $\bigwedge x\ y\ z.\ x \cdot (d\ y \cdot z) = (x \cdot d\ y\ ) \cdot z$
shows $\nu\ ((\nu\ x)^\omega) = \nabla\ (\nu\ x) \cdot nc$
⟨*proof*⟩

lemma *tau-om-nu*:
assumes $\bigwedge x\ y\ z.\ x \cdot (d\ y \cdot z) = (x \cdot d\ y\ ) \cdot z$
shows $\tau\ ((\nu\ x)^\omega) = \nabla\ (\nu\ x) \cdot 1_\pi$
⟨*proof*⟩

Proposition 13.12.

lemma *wf-eq-defl*: $(\forall\, y.\ d\ y \leq d\ (x \cdot y) \longrightarrow d\ y = 0) \longleftrightarrow (\forall\, y.\ y \leq x \cdot y \longrightarrow y = 0)$
⟨*proof*⟩

lemma *defl-eq-om-trivial*: $x^\omega = 0 \longleftrightarrow (\forall\, y.\ y \leq x \cdot y \longrightarrow y = 0)$
⟨*proof*⟩

lemma *wf-eq-om-trivial*: $x^\omega = 0 \longleftrightarrow (\forall\, y.\ d\ y \leq d\ (x \cdot y) \longrightarrow d\ y = 0)$
⟨*proof*⟩

end

2.12 Proto-Quantales

Finally we define the class of proto-quantales and prove some of the remaining facts from the article. Full c-quantales, as defined there, are not needed

for these proofs.

class *proto-quantale* = *complete-lattice* + *proto-monoid* +
  assumes *Sup-mult-distr*: $Sup\ X \cdot y = Sup\ \{x \cdot y \mid x.\ x \in X\}$
  and *isol*: $x \leq y \Longrightarrow z \cdot x \leq z \cdot y$

begin

sublocale *pd?*: *proto-dioid* 1_σ $(\cdot)$ *sup* $(\leq)$ $(<)$ *Sup* $\{\}$
⟨*proof*⟩

definition *star-rd* :: $'a \Rightarrow 'a$ **where**
  $star\text{-}rd\ x = Sup\ \{power\text{-}rd\ x\ i\ |i.\ i \in \mathbb{N}\}$

definition *star-sq* :: $'a \Rightarrow 'a$ **where**
  $star\text{-}sq\ x = Sup\ \{power\text{-}sq\ x\ i\ |i.\ i \in \mathbb{N}\}$

Now we prove Lemma 12.6.

lemma *star-rd-le-sq*: $star\text{-}rd\ x \leq star\text{-}sq\ x$
  ⟨*proof*⟩

lemma *star-sq-le-rd*: $star\text{-}sq\ x \leq star\text{-}rd\ x$
  ⟨*proof*⟩

lemma *star-rd-sq*: $star\text{-}rd\ x = star\text{-}sq\ x$
  ⟨*proof*⟩

lemma *star-sq-power*: $star\text{-}sq\ x = Sup\ \{pd.p\text{-}power\ (sup\ 1_\sigma\ x)\ i \mid i.\ i \in \mathbb{N}\}$
  ⟨*proof*⟩

The following lemma should be somewhere close to complete lattices.

end

lemma *mono-aux*: *mono* $(\lambda y.\ sup\ (z{::}\ 'a :: proto\text{-}quantale)\ (x \cdot y))$
  ⟨*proof*⟩

lemma *gfp-lfp-prop*: $sup\ (gfp\ (\lambda(y :: 'a :: proto\text{-}quantale).\ x \cdot y))\ (lfp\ (\lambda y.\ sup\ z\ (x \cdot y))) \leq gfp\ (\lambda y.\ sup\ z\ \ (x \cdot y))$
  ⟨*proof*⟩

end

3 Multirelations

theory *Multirelations*
imports *C-Algebras*
begin

3.1 Basic Definitions

We define a type synonym for multirelations.

type-synonym *'a mrel* = (*'a* ∗ (*'a set*)) *set*

no-notation *s-prod* (**infixl** · *80*)
no-notation *s-id* (1_σ)
no-notation *c-prod* (**infixl** ‖ *80*)
no-notation *c-id* (1_π)

Now we start with formalising the multirelational model. First we define sequential composition and paraellel composition of multirelations, their units and the universal multirelation as in Section 2 of the article.

definition *s-prod* :: *'a mrel* ⇒ *'a mrel* ⇒ *'a mrel* (**infixl** · *70*) **where**
R · *S* = {(*a*,*A*). (∃ *B*. (*a*,*B*) ∈ *R* ∧ (∃ *f*. (∀ *b* ∈ *B*. (*b*,*f b*) ∈ *S*) ∧ *A* = ⋃{*f b* |*b*. *b* ∈ *B*}))}

definition *s-id* :: *'a mrel* (1_σ) **where**
1_σ ≡ ⋃ *a*. {(*a*,{*a*})}

definition *p-prod* :: *'a mrel* ⇒ *'a mrel* ⇒ *'a mrel* (**infixl** ‖ *70*) **where**
R ‖ *S* = {(*a*,*A*). (∃ *B C*. *A* = *B* ∪ *C* ∧ (*a*,*B*) ∈ *R* ∧ (*a*,*C*) ∈ *S*)}

definition *p-id* :: *'a mrel* (1_π) **where**
1_π ≡ ⋃ *a*. {(*a*,{})}

definition *U* :: *'a mrel* **where**
U ≡ {(*a*,*A*) |*a A*. *a* ∈ *UNIV* ∧ *A* ⊆ *UNIV*}

abbreviation *NC* ≡ *U* − 1_π

We write NC where $\overline{1_\pi}$ is written in [2].

Next we prove some basic set-theoretic properties.

lemma *s-prod-im*: *R* · *S* = {(*a*,*A*). (∃ *B*. (*a*,*B*) ∈ *R* ∧ (∃ *f*. (∀ *b* ∈ *B*. (*b*,*f b*) ∈ *S*) ∧ *A* = ⋃((λ*x*. *f x*) ‘ *B*)))}
⟨*proof*⟩

lemma *s-prod-iff*: (*a*,*A*) ∈ (*R* · *S*) ⟷ (∃ *B*. (*a*,*B*) ∈ *R* ∧ (∃ *f*. (∀ *b* ∈ *B*. (*b*,*f b*) ∈ *S*) ∧ *A* = ⋃((λ*x*. *f x*) ‘ *B*)))
⟨*proof*⟩

lemma *s-id-iff*: (*a*,*A*) ∈ 1_σ ⟷ *A* = {*a*}
⟨*proof*⟩

lemma *p-prod-iff*: (*a*,*A*) ∈ *R* ‖ *S* ⟷ (∃ *B C*. *A* = *B* ∪ *C* ∧ (*a*,*B*) ∈ *R* ∧ (*a*,*C*) ∈ *S*)
⟨*proof*⟩

named-theorems *mr-simp*
declare *s-prod-im* [*mr-simp*] *p-prod-def* [*mr-simp*] *s-id-def* [*mr-simp*] *p-id-def* [*mr-simp*] *U-def* [*mr-simp*]

3.2 Multirelations and Proto-Dioids

We can now show that multirelations form proto-trioids. This is Proposition 5.1, and it subsumes Proposition 4.1,

interpretation *mrel-proto-trioid*: *proto-trioid* 1_σ $(\cdot)$ 1_π $(\|)$ $(\cup)$ $(\subseteq)$ $(\subset)$ $\{\}$
⟨*proof*⟩

3.3 Simple Properties

This covers all the identities in the display before Lemma 2.1 except the two following ones.

lemma *s-prod-assoc1*: $(R \cdot S\ ) \cdot T \subseteq R \cdot (S \cdot T)$
⟨*proof*⟩

lemma *seq-conc-subdistr*: $(R \parallel S) \cdot T \subseteq (R \cdot T) \parallel (S \cdot T)$
⟨*proof*⟩

Next we provide some counterexamples. These do not feature in [2].

lemma $R \cdot \{\} = \{\}$
nitpick
⟨*proof*⟩

lemma $R \cdot (S \cup T) = R \cdot S \cup R \cdot T$
⟨*proof*⟩

lemma $R \cdot (S \cdot T) \subseteq (R \cdot S) \cdot T$
⟨*proof*⟩

lemma $(R \parallel R) \cdot T = (R \cdot T) \parallel (R \cdot T)$
quickcheck
⟨*proof*⟩

Next we prove the distributivity and associativity laws for sequential subidentities mentioned before Lemma 2.1

lemma *subid-aux2*:
assumes $R \subseteq 1_\sigma$**and** $(a,A) \in R$
shows $A = \{a\}$
⟨*proof*⟩

lemma *s-prod-test-aux1*:
assumes $S \subseteq 1_\sigma$
and $(a,A) \in R \cdot S$
shows $((a,A) \in R \wedge (\forall\, a \in A.\ (a,\{a\}) \in S))$

⟨*proof*⟩

lemma *s-prod-test-aux2*:
assumes $(a,A) \in R$
and $\forall\, a \in A.\ (a,\{a\}) \in S$
shows $(a,A) \in R \cdot S$
⟨*proof*⟩

lemma *s-prod-test*:
assumes $P \subseteq 1_\sigma$
shows $(a,A) \in R \cdot P \longleftrightarrow (a,A) \in R \wedge (\forall\, a \in A.\ (a,\{a\}) \in P)$
⟨*proof*⟩

lemma *test-s-prod-aux1*:
assumes $P \subseteq 1_\sigma$
and $(a,A) \in P \cdot R$
shows $(a,\{a\}) \in P \wedge (a,A) \in R$
⟨*proof*⟩

lemma *test-s-prod-aux2*:
assumes $(a,A) \in R$
and $(a,\{a\}) \in P$
shows $(a,A) \in P \cdot R$
⟨*proof*⟩

lemma *test-s-prod*:
assumes $P \subseteq 1_\sigma$
shows $(a,A) \in P \cdot R \longleftrightarrow (a,\{a\}) \in P \wedge (a,A) \in R$
⟨*proof*⟩

lemma *test-assoc1*:
assumes $P \subseteq 1_\sigma$
shows $(R \cdot P) \cdot S = R \cdot (P \cdot S)$
⟨*proof*⟩

lemma *test-assoc2*:
assumes $P \subseteq 1_\sigma$
shows $(P \cdot R) \cdot S = P \cdot (R \cdot S)$
⟨*proof*⟩

lemma *test-assoc3*:
assumes $P \subseteq 1_\sigma$
shows $(R \cdot S) \cdot P = R \cdot (S \cdot P)$
⟨*proof*⟩

lemma *s-distl-test*:
assumes $R \subseteq 1_\sigma$
shows $R \cdot (S \cup T) = R \cdot S \cup R \cdot T$
⟨*proof*⟩

Next we verify Lemma 2.1.

lemma *subid-par-idem*:
assumes $R \subseteq 1_\sigma$
shows $R \parallel R = R$
⟨*proof*⟩

lemma *term-par-idem*:
assumes $R \subseteq 1_\pi$
shows $R \parallel R = R$
⟨*proof*⟩

lemma *U-par-idem*: $U \parallel U = U$
⟨*proof*⟩

lemma *nc-par-idem*: $NC \parallel NC = NC$
⟨*proof*⟩

Next we prove the properties of Lemma 2.2 and 3.2. First we prepare to show that multirelations form c-lattices.

We define the domain operation on multirelations and verify the explicit definition from Section 3.

definition $d :: 'a\ mrel \Rightarrow 'a\ mrel$ **where**
$d\ R \equiv \{(a,\{a\})\ |a.\ \exists B.\ (a,B) \in R\}$

named-theorems *mrd-simp*
declare *mr-simp* [*mrd-simp*] *d-def* [*mrd-simp*]

lemma *d-def-expl*: $d\ R = (R \cdot 1_\pi) \parallel 1_\sigma$
⟨*proof*⟩

interpretation *mrel-pbdl-monoid*: *pbdl-monoid* 1_σ $(\cdot)$ 1_π $(\parallel)$ $(\cup)$ $(\subseteq)$ $(\subset)$ $\{\}$ U $(\cap)$
⟨*proof*⟩

Here come the properties of Lemma 2.2.

lemma *c1*: $(R \cdot 1_\pi) \parallel R = R$
⟨*proof*⟩

lemma *t-aux*: $T \parallel T \subseteq T \Longrightarrow (\forall\, a\ B\ C.\ (a,B) \in T \wedge (a,C) \in T \Longrightarrow (a,B \cup C) \in T)$
⟨*proof*⟩

lemma *cl4*:
assumes $T \parallel T \subseteq T$
shows $(R \cdot T) \parallel (S \cdot T) \subseteq (R \parallel S) \cdot T$
⟨*proof*⟩

lemma *cl3*: $R \cdot (S \parallel T) \subseteq (R \cdot S) \parallel (R \cdot T)$
⟨*proof*⟩

lemma *cl5*: $(R \cdot S) \cdot (T \cdot \{\}) = R \cdot (S \cdot (T \cdot \{\}))$
⟨*proof*⟩

We continue verifying other c-lattice axioms

lemma *cl8-var*: $d\ R \cdot S = (R \cdot 1_\pi) \parallel S$
⟨*proof*⟩

lemma *cl9-var*: $d\ (R \cap 1_\sigma) = R \cap 1_\sigma$
⟨*proof*⟩

lemma *cl10-var*: $d\ (R - 1_\pi) = 1_\sigma \cap ((R - 1_\pi) \cdot NC)$
⟨*proof*⟩

3.4 Multirelations and C-Lattices

Next we show that multirelations form c-lattices (Proposition 7.3) and prove further facts in this setting.

interpretation *mrel-c-lattice*: *c-lattice* 1_σ $(\cdot)$ 1_π $(\parallel)$ $(\cup)$ $(\subseteq)$ $(\subset)$ $\{\}$ U $(\cap)$ NC
⟨*proof*⟩

The following facts from Lemma 2.2 remain to be shown.

lemma *p-id-assoc1*: $(1_\pi \cdot R) \cdot S = 1_\pi \cdot (R \cdot S)$
⟨*proof*⟩

lemma *p-id-assoc2*: $(R \cdot 1_\pi) \cdot T = R \cdot (1_\pi \cdot T)$
⟨*proof*⟩

lemma *seq-conc-subdistrl*:
assumes $P \subseteq 1_\sigma$
shows $P \cdot (S \parallel T) = (P \cdot S) \parallel (P \cdot T)$
⟨*proof*⟩

lemma *test-s-prod-is-meet* [*simp*]:
assumes $R \subseteq 1_\sigma$
and $S \subseteq 1_\sigma$
shows $R \cdot S = R \cap S$
⟨*proof*⟩

lemma *test-p-prod-is-meet*:
assumes $R \subseteq 1_\sigma$
and $S \subseteq 1_\sigma$
shows $R \parallel S = R \cap S$
⟨*proof*⟩

lemma *test-multipliciativer*:

assumes $R \subseteq 1_\sigma$
and $S \subseteq 1_\sigma$
shows $(R \cap S) \cdot T = (R \cdot T) \cap (S \cdot T)$
⟨*proof*⟩

Next we verify the remaining fact from Lemma 2.2; in fact it follows from the corresponding theorem of c-lattices.

lemma *c6*: $R \cdot 1_\pi \subseteq 1_\pi$
⟨*proof*⟩

Next we verify Lemma 3.1.

lemma *p-id-st*: $R \cdot 1_\pi = \{(a,\{\}) \mid a.\ \exists B.\ (a,B) \in R\}$
⟨*proof*⟩

lemma *p-id-zero*: $R \cap 1_\pi = R \cdot \{\}$
⟨*proof*⟩

lemma *p-id-zero-st*: $R \cap 1_\pi = \{(a,\{\}) \mid a.\ (a,\{\}) \in R\}$
⟨*proof*⟩

lemma *s-id-st*: $R \cap 1_\sigma = \{(a,\{a\}) \mid a.\ (a,\{a\}) \in R\}$
⟨*proof*⟩

lemma *U-seq-st*: $(a,A) \in R \cdot U \longleftrightarrow (A = \{\} \wedge (a,\{\}) \in R) \vee (\exists B.\ B \neq \{\} \wedge (a,B) \in R)$
⟨*proof*⟩

lemma *U-par-st*: $(a,A) \in R \parallel U \longleftrightarrow (\exists B.\ B \subseteq A \wedge (a,B) \in R)$
⟨*proof*⟩

Next we verify the relationships after Lemma 3.1.

lemma *s-subid-iff1*: $R \subseteq 1_\sigma \longleftrightarrow R \cap 1_\sigma = R$
⟨*proof*⟩

lemma *s-subid-iff2*: $R \subseteq 1_\sigma \longleftrightarrow d\ R = R$
⟨*proof*⟩

lemma *p-subid-iff*: $R \subseteq 1_\pi \longleftrightarrow R \cdot 1_\pi = R$
⟨*proof*⟩

lemma *vec-iff1*:
assumes $\forall a.\ (\exists A.\ (a,A) \in R) \longrightarrow (\forall A.\ (a,A) \in R)$
shows $(R \cdot 1_\pi) \parallel U = R$
⟨*proof*⟩

lemma *vec-iff2*:
assumes $(R \cdot 1_\pi) \parallel U = R$
shows $(\forall a.\ (\exists A.\ (a,A) \in R) \longrightarrow (\forall A.\ (a,A) \in R))$
⟨*proof*⟩

lemma *vec-iff*: $(\forall\, a.\ (\exists\, A.\ (a,A) \in R) \longrightarrow (\forall\, A.\ (a,A) \in R)) \longleftrightarrow (R \cdot 1_\pi) \parallel U = R$
⟨*proof*⟩

lemma *ucl-iff*: $(\forall\, a\ A\ B.\ (a,A) \in R \wedge A \subseteq B \longrightarrow (a,B) \in R) \longleftrightarrow R \parallel U = R$
⟨*proof*⟩

lemma *nt-iff*: $R \subseteq NC \longleftrightarrow R \cap NC = R$
⟨*proof*⟩

Next we provide a counterexample for the final paragraph of Section 3.

lemma $1_\sigma \cap R \cdot U = R$
nitpick
⟨*proof*⟩

Next we present a counterexample for vectors mentioned before Lemma 9.3.

lemma $d\ (d\ R \cdot U) \cdot (d\ S \cdot U) \cdot U = (d\ R \cdot U) \cdot (d\ S \cdot U)$
nitpick
⟨*proof*⟩

Next we prove Tarski' rule (Lemma 9.3).

lemma *tarski-aux*:
assumes $R - 1_\pi \neq \{\}$
and $(a,A) \in NC$
shows $(a,A) \in NC \cdot ((R - 1_\pi) \cdot NC)$
⟨*proof*⟩

lemma *tarski*:
assumes $R - 1_\pi \neq \{\}$
shows $NC \cdot ((R - 1_\pi) \cdot NC) = NC$
⟨*proof*⟩

Next we verify the assumptions of Proposition 9.8.

lemma *d-assoc1*: $d\ R \cdot (S \cdot T) = (d\ R \cdot S) \cdot T$
⟨*proof*⟩

lemma *d-meet-distr-var*: $(d\ R \cap d\ S) \cdot T = (d\ R \cdot T) \cap (d\ S \cdot T)$
⟨*proof*⟩

Lemma 10.5.

lemma $((R \cap 1_\sigma) \cdot (S \cap 1_\sigma)) \cdot 1_\pi = ((R \cap 1_\sigma) \cdot 1_\pi) \cdot ((S \cap 1_\sigma) \cdot 1_\pi)$
nitpick
⟨*proof*⟩

lemma $d\ ((R \cdot 1_\pi) \cdot (S \cdot 1_\pi)) = d\ (R \cdot 1_\pi) \cdot d\ (S \cdot 1_\pi)$
nitpick
⟨*proof*⟩

lemma $((R \cap 1_\sigma) \cdot (S \cap 1_\sigma)) \cdot U = ((R \cap 1_\sigma) \cdot U) \cdot ((S \cap 1_\sigma) \cdot U)$
nitpick
⟨*proof*⟩

lemma $d\ (((R \cdot 1_\pi) \parallel U) \cdot ((S \cdot 1_\pi) \parallel U)) = d\ ((R \cdot 1_\pi) \parallel U) \cdot d\ ((S \cdot 1_\pi) \parallel U)$
nitpick
⟨*proof*⟩

lemma $((R \cdot 1_\pi) \cdot (S \cdot 1_\pi)) \parallel U = ((R \cdot 1_\pi) \parallel U) \cdot ((S \cdot 1_\pi) \parallel U)$
nitpick
⟨*proof*⟩

lemma $(((R - 1_\pi) \cap 1_\sigma) \cdot ((S - 1_\pi) \cap 1_\sigma)) \cdot 1_\pi = (((R - 1_\pi) \cap 1_\sigma) \cdot 1_\pi) \cdot (((S - 1_\pi) \cap 1_\sigma) \cdot 1_\pi)$
nitpick
⟨*proof*⟩

lemma $d\ (((R - 1_\pi) \cdot 1_\pi) \cdot ((S - 1_\pi) \cdot 1_\pi)) = d\ ((R - 1_\pi) \cdot 1_\pi) \cdot d\ ((S - 1_\pi) \cdot 1_\pi)$
nitpick
⟨*proof*⟩

lemma $(((R - 1_\pi) \cap 1_\sigma) \cdot ((S - 1_\pi) \cap 1_\sigma)) \cdot NC = (((R - 1_\pi) \cap 1_\sigma) \cdot NC) \cdot (((S - 1_\pi) \cap 1_\sigma) \cdot NC)$
⟨*proof*⟩

lemma $d\ ((((R - 1_\pi) \cdot 1_\pi) \parallel NC) \cdot (((S - 1_\pi) \cdot 1_\pi) \parallel NC)) = d\ (((R - 1_\pi) \cdot 1_\pi) \parallel NC) \cdot d\ (((S - 1_\pi) \cdot 1_\pi) \parallel NC)$
⟨*proof*⟩

lemma $(((R - 1_\pi) \cdot 1_\pi) \cdot ((S - 1_\pi) \cdot 1_\pi)) \parallel NC = (((R - 1_\pi) \cdot 1_\pi) \parallel NC) \cdot (((S - 1_\pi) \cdot 1_\pi) \parallel NC)$
nitpick
⟨*proof*⟩

lemma $((((R - 1_\pi) \cdot 1_\pi) \parallel NC) \cdot (((S - 1_\pi) \cdot 1_\pi) \parallel NC)) \cdot 1_\pi = ((((R - 1_\pi) \cdot 1_\pi) \parallel NC) \cdot 1_\pi) \cdot ((((S - 1_\pi) \cdot 1_\pi) \parallel NC) \cdot 1_\pi)$
nitpick
⟨*proof*⟩

3.5 Terminal and Nonterminal Elements

Lemma 11.4

lemma $(R \cdot S) \cdot \{\} = (R \cdot \{\}) \cdot (S \cdot \{\})$
nitpick
⟨*proof*⟩

lemma $(R \cdot S) - 1_\pi = (R - 1_\pi) \cdot (S - 1_\pi)$

⟨*proof*⟩

lemma $(R \parallel S) - 1_\pi = (R - 1_\pi) \parallel (S - 1_\pi)$
nitpick
⟨*proof*⟩

Lemma 11.8.

lemma $((R \cdot 1_\pi) \cdot (S - 1_\pi)) - 1_\pi = (R \cdot 1_\pi) \cdot (S - 1_\pi)$
nitpick
⟨*proof*⟩

lemma $((S - 1_\pi) \cdot (R \cdot 1_\pi)) - 1_\pi = (S - 1_\pi) \cdot (R \cdot 1_\pi)$
nitpick
⟨*proof*⟩

lemma $((R \cdot 1_\pi) \parallel (S - 1_\pi)) \cdot 1_\pi = (R \cdot 1_\pi) \parallel (S - 1_\pi)$
nitpick
⟨*proof*⟩

Lemma 11.10.

lemma $R \cdot \{\} \subseteq S \cdot \{\} \Longrightarrow (R \cdot T) \cdot \{\} \subseteq (S \cdot T) \cdot \{\}$
nitpick
⟨*proof*⟩

lemma $R - 1_\pi \subseteq S - 1_\pi \Longrightarrow (R \parallel T) - 1_\pi \subseteq (S \parallel T) - 1_\pi$
nitpick
⟨*proof*⟩

lemma $R - 1_\pi \subseteq S - 1_\pi \Longrightarrow (T \cdot R) - 1_\pi \subseteq (T \cdot S) - 1_\pi$
⟨*proof*⟩

Corollary 11.12

lemma $R \cdot \{\} = S \cdot \{\} \Longrightarrow (R \cdot T) \cdot \{\} = (S \cdot T) \cdot \{\}$
nitpick
⟨*proof*⟩

lemma $R - 1_\pi = S - 1_\pi \Longrightarrow (R \parallel T) - 1_\pi = (S \parallel T) - 1_\pi$
nitpick
⟨*proof*⟩

lemma $R - 1_\pi = S - 1_\pi \Longrightarrow (T \cdot R) - 1_\pi = (T \cdot S) - 1_\pi$
⟨*proof*⟩

3.6 Multirelations, Proto-Quantales and Iteration

interpretation *mrel-proto-quantale*: *proto-quantale* 1_σ $(\cdot)$ *Inter Union* $(\cap)$ $(\subseteq)$ $(\subset)$ $(\cup)$ $\{\}$ *U*
⟨*proof*⟩

We reprove Corollary 13.2. because Isabelle does not pick it up from the quantale level.

lemma *iso-prop*: *mono* $(\lambda X.\ S \cup R \cdot X)$
⟨*proof*⟩

lemma *gfp-lfp-prop*: *gfp* $(\lambda X.\ R \cdot X) \cup$ *lfp* $(\lambda X.\ S \cup R \cdot X) \subseteq$ *gfp* $(\lambda X.\ S \cup R \cdot X)$
⟨*proof*⟩

3.7 Further Counterexamples

Lemma 14,1. and 14.2

lemma $R \parallel R \subseteq R$
nitpick
⟨*proof*⟩

lemma $R \subseteq R \parallel S$
nitpick
⟨*proof*⟩

lemma $R \parallel S \cap R \parallel T \subseteq R \parallel (S \cap T)$
nitpick
⟨*proof*⟩

lemma $R \cdot (S \parallel T) = (R \cdot S) \parallel (R \cdot T)$
nitpick
⟨*proof*⟩

lemma $R \cdot (S \cdot T) \subseteq (R \cdot S) \cdot T$
⟨*proof*⟩

lemma $[\![R \parallel R = R;\ S \parallel S = S;\ T \parallel T = T]\!] \Longrightarrow R \cdot (S \parallel T) = (R \cdot S) \parallel (R \cdot T)$
nitpick
⟨*proof*⟩

lemma $[\![R \neq \{\};\ S \neq \{\};\ \forall a.\ (a,\{\}) \notin R \cup S]\!] \Longrightarrow R \cdot S \subseteq R \parallel S$
quickcheck
⟨*proof*⟩

lemma $[\![R \neq \{\};\ S \neq \{\};\ \forall a.\ (a,\{\}) \notin R \cup S]\!] \Longrightarrow R \parallel S \subseteq R \cdot S$
quickcheck
⟨*proof*⟩

lemma $[\![R \neq \{\};\ S \neq \{\};\ T \neq \{\};\ \forall a.\ (a,\{\}) \notin R \cup S \cup T]\!] \Longrightarrow (R \parallel S) \cdot T \subseteq R \parallel (S \cdot T)$
quickcheck
⟨*proof*⟩

lemma $[\![R \neq \{\};\ S \neq \{\};\ T \neq \{\};\ \forall\, a.\ (a,\{\}) \notin R \cup S \cup T]\!] \Longrightarrow R \parallel (S \cdot T) \subseteq (R \parallel S) \cdot T$
quickcheck
⟨*proof*⟩

lemma $[\![R \neq \{\};\ S \neq \{\};\ T \neq \{\};\ \forall\, a.\ (a,\{\}) \notin R \cup S \cup T]\!] \Longrightarrow R \cdot (S \parallel T) \subseteq (R \cdot S) \parallel T$
quickcheck
⟨*proof*⟩

lemma $[\![R \neq \{\};\ S \neq \{\};\ T \neq \{\};\ \forall\, a.\ (a,\{\}) \notin R \cup S \cup T]\!] \Longrightarrow (R \cdot S) \parallel T \subseteq R \cdot (S \parallel T)$
quickcheck
⟨*proof*⟩

lemma $[\![R \neq \{\};\ S \neq \{\};\ \forall\, a.\ (a,\{\}) \notin R \cup S]\!] \Longrightarrow (R \parallel S) \cdot (R \parallel S) \subseteq (R \cdot R) \parallel (S \cdot S)$
quickcheck
⟨*proof*⟩

lemma $[\![R \neq \{\};\ S \neq \{\};\ \forall\, a.\ (a,\{\}) \notin R \cup S]\!] \Longrightarrow (R \cdot R) \parallel (S \cdot S) \subseteq (R \parallel S) \cdot (R \parallel S)$
quickcheck
⟨*proof*⟩

3.8 Relationship with Up-Closed Multirelations

We now define Parikh's sequential composition.

definition *s-prod-pa* :: *'a mrel* ⇒ *'a mrel* ⇒ *'a mrel* (**infixl** ⊗ *70*) **where**
$R \otimes S = \{(a,A).\ (\exists\, B.\ (a,B) \in R \wedge (\forall\, b \in B.\ (b,A) \in S))\}$

We show that Parikh's definition doesn't preserve up-closure.

lemma *up-closed-prop*: $((R \parallel U) \cdot (S \parallel U)) \parallel U = (R \parallel U) \cdot (S \parallel U)$
⟨*proof*⟩

Lemma 15.1.

lemma *onelem*: $(R \cdot S) \parallel U \subseteq R \otimes (S \parallel U)$
⟨*proof*⟩

lemma *twolem*: $R \otimes (S \parallel U) \subseteq (R \cdot S) \parallel U$
⟨*proof*⟩

lemma *pe-pa-sim*: $(R \cdot S) \parallel U = R \otimes (S \parallel U)$
⟨*proof*⟩

lemma *pe-pa-sim-var*: $((R \parallel U) \cdot (S \parallel U)) \parallel U = (R \parallel U) \otimes (S \parallel U)$
⟨*proof*⟩

lemma *pa-assoc1*: $((R \parallel U) \otimes (S \parallel U)) \otimes (T \parallel U) \subseteq (R \parallel U) \otimes ((S \parallel U) \otimes (T \parallel U))$
⟨*proof*⟩

The converse direction of associativity remains to be proved.

Corollary 15.3.

lemma *up-closed-par-is-meet*: $(R \parallel U) \parallel (S \parallel U) = (R \parallel U) \cap (S \parallel U)$
⟨*proof*⟩

end

References

[1] H. Furusawa and G. Struth. Concurrent dynamic algebra. *ACM Transactions on Computational Logic*, 2015. (In Press).

[2] H. Furusawa and G. Struth. Taming multirelations. *CoRR*, abs/1501.05147, 2015.

[3] D. Peleg. Concurrent dynamic logic. *J. ACM*, 34(2):450–479, 1987.