On Domain Semigroups, Stably Supported Quantales and Frames

Peter Jipsen

Chapman University

Dec 4, 2009

Outline

- Domain and antidomain operations on relations
- Motivation for studying domain semigroups
- Stably supported quantales and frames
- Twisted domain monoids and representability
- Hirsch-Mikulas nonrepresentability result
- Free domain monoids
- Conclusion

$Rel(X)$ = set of *binary relations* R on a set X

$R;S$ = *composition* of relations $= \{(u,v) : \exists w\, (u,w) \in R, (w,v) \in S\}$

unary *domain*, *antidomain*, *range* and *converse* operations are defined by

$$d(R) = \{(u,u) \in X^2 : (u,v) \in R \text{ for some } v \in X\}$$

$$a(R) = \{(u,u) \in X^2 : (u,v) \notin R \text{ for all } v \in X\}$$

$$r(R) = \{(v,v) \in X^2 : (u,v) \in R \text{ for some } u \in X\}$$

$$R^{\smile} = \{(v,u) \in X^2 : (u,v) \in R\}$$

In *relation algebras*, $d(R) = R;R^{\smile} \cap \mathrm{id}$, $a(R) = \mathrm{id} - d(R)$,

$r(R) = R^{\smile};R \cap \mathrm{id}$, $\mathrm{id} = \{(u,u) : u \in X\}$

$Rel(X)$ = set of *binary relations* R on a set X

$R;S$ = *composition* of relations $= \{(u,v) : \exists w\,(u,w) \in R, (w,v) \in S\}$

unary *domain*, *antidomain*, *range* and *converse* operations are defined by

$$d(R) = \{(u,u) \in X^2 : (u,v) \in R \text{ for some } v \in X\}$$

$$a(R) = \{(u,u) \in X^2 : (u,v) \notin R \text{ for all } v \in X\}$$

$$r(R) = \{(v,v) \in X^2 : (u,v) \in R \text{ for some } u \in X\}$$

$$R^{\smile} = \{(v,u) \in X^2 : (u,v) \in R\}$$

In *relation algebras*, $d(R) = R;R^{\smile} \cap \mathrm{id}$, $a(R) = \mathrm{id} - d(R)$,

$r(R) = R^{\smile};R \cap \mathrm{id}$, $\mathrm{id} = \{(u,u) : u \in X\}$

$Rel(X)$ = set of *binary relations* R on a set X

$R;S$ = *composition* of relations $= \{(u,v) : \exists w\,(u,w) \in R, (w,v) \in S\}$

unary *domain*, *antidomain*, *range* and *converse* operations are defined by

$$d(R) = \{(u,u) \in X^2 : (u,v) \in R \text{ for some } v \in X\}$$

$$a(R) = \{(u,u) \in X^2 : (u,v) \notin R \text{ for all } v \in X\}$$

$$r(R) = \{(v,v) \in X^2 : (u,v) \in R \text{ for some } u \in X\}$$

$$R^{\smile} = \{(v,u) \in X^2 : (u,v) \in R\}$$

In *relation algebras*, $d(R) = R;R^{\smile} \cap \mathrm{id}$, $a(R) = \mathrm{id} - d(R)$,

$r(R) = R^{\smile};R \cap \mathrm{id}$, $\mathrm{id} = \{(u,u) : u \in X\}$

$Rel(X)$ = set of *binary relations* R on a set X

$R;S$ = *composition* of relations $= \{(u,v) : \exists w\,(u,w) \in R, (w,v) \in S\}$

unary *domain*, *antidomain*, *range* and *converse* operations are defined by

$$d(R) = \{(u,u) \in X^2 : (u,v) \in R \text{ for some } v \in X\}$$

$$a(R) = \{(u,u) \in X^2 : (u,v) \notin R \text{ for all } v \in X\}$$

$$r(R) = \{(v,v) \in X^2 : (u,v) \in R \text{ for some } u \in X\}$$

$$R^{\smile} = \{(v,u) \in X^2 : (u,v) \in R\}$$

In *relation algebras*, $d(R) = R;R^{\smile} \cap \mathrm{id}$, $a(R) = \mathrm{id} - d(R)$,

$r(R) = R^{\smile};R \cap \mathrm{id}$, $\mathrm{id} = \{(u,u) : u \in X\}$

$Rel(X)$ = set of *binary relations* R on a set X

$R;S$ = *composition* of relations $= \{(u,v) : \exists w\,(u,w) \in R, (w,v) \in S\}$

unary *domain*, *antidomain*, *range* and *converse* operations are defined by

$$d(R) = \{(u,u) \in X^2 : (u,v) \in R \text{ for some } v \in X\}$$

$$a(R) = \{(u,u) \in X^2 : (u,v) \notin R \text{ for all } v \in X\}$$

$$r(R) = \{(v,v) \in X^2 : (u,v) \in R \text{ for some } u \in X\}$$

$$R^{\smile} = \{(v,u) \in X^2 : (u,v) \in R\}$$

In *relation algebras*, $d(R) = R;R^{\smile} \cap \mathrm{id}$, $a(R) = \mathrm{id} - d(R)$,

$r(R) = R^{\smile};R \cap \mathrm{id}$, $\mathrm{id} = \{(u,u) : u \in X\}$

$Rel(X)$ = set of *binary relations* R on a set X

$R;S$ = *composition* of relations $= \{(u,v) : \exists w\,(u,w) \in R, (w,v) \in S\}$

unary *domain*, *antidomain*, *range* and *converse* operations are defined by

$$d(R) = \{(u,u) \in X^2 : (u,v) \in R \text{ for some } v \in X\}$$

$$a(R) = \{(u,u) \in X^2 : (u,v) \notin R \text{ for all } v \in X\}$$

$$r(R) = \{(v,v) \in X^2 : (u,v) \in R \text{ for some } u \in X\}$$

$$R^{\smile} = \{(v,u) \in X^2 : (u,v) \in R\}$$

In *relation algebras*, $d(R) = R;R^{\smile} \cap \mathrm{id}$, $a(R) = \mathrm{id} - d(R)$,

$r(R) = R^{\smile};R \cap \mathrm{id}$, $\mathrm{id} = \{(u,u) : u \in X\}$

$Rel(X)$ = set of *binary relations* R on a set X

$R;S$ = *composition* of relations $= \{(u,v) : \exists w\,(u,w) \in R, (w,v) \in S\}$

unary *domain*, *antidomain*, *range* and *converse* operations are defined by

$$d(R) = \{(u,u) \in X^2 : (u,v) \in R \text{ for some } v \in X\}$$

$$a(R) = \{(u,u) \in X^2 : (u,v) \notin R \text{ for all } v \in X\}$$

$$r(R) = \{(v,v) \in X^2 : (u,v) \in R \text{ for some } u \in X\}$$

$$R^{\smile} = \{(v,u) \in X^2 : (u,v) \in R\}$$

In *relation algebras*, $d(R) = R;R^{\smile} \cap \mathrm{id}$, $a(R) = \mathrm{id} - d(R)$,

$r(R) = R^{\smile};R \cap \mathrm{id}$, $\mathrm{id} = \{(u,u) : u \in X\}$

$Rel(X)$ = set of *binary relations* R on a set X

$R;S$ = *composition* of relations $= \{(u,v) : \exists w\,(u,w) \in R, (w,v) \in S\}$

unary *domain*, *antidomain*, *range* and *converse* operations are defined by

$$d(R) = \{(u,u) \in X^2 : (u,v) \in R \text{ for some } v \in X\}$$

$$a(R) = \{(u,u) \in X^2 : (u,v) \notin R \text{ for all } v \in X\}$$

$$r(R) = \{(v,v) \in X^2 : (u,v) \in R \text{ for some } u \in X\}$$

$$R^{\smile} = \{(v,u) \in X^2 : (u,v) \in R\}$$

In *relation algebras*, $d(R) = R;R^{\smile} \cap \mathrm{id}$, $a(R) = \mathrm{id} - d(R)$,

$r(R) = R^{\smile};R \cap \mathrm{id}$, $\mathrm{id} = \{(u,u) : u \in X\}$

$Rel(X)$ = set of *binary relations* R on a set X

$R;S$ = *composition* of relations $= \{(u,v) : \exists w\,(u,w) \in R, (w,v) \in S\}$

unary *domain*, *antidomain*, *range* and *converse* operations are defined by

$$d(R) = \{(u,u) \in X^2 : (u,v) \in R \text{ for some } v \in X\}$$

$$a(R) = \{(u,u) \in X^2 : (u,v) \notin R \text{ for all } v \in X\}$$

$$r(R) = \{(v,v) \in X^2 : (u,v) \in R \text{ for some } u \in X\}$$

$$R^{\smile} = \{(v,u) \in X^2 : (u,v) \in R\}$$

In *relation algebras*, $\quad d(R) = R;R^{\smile} \cap \mathrm{id}, \qquad a(R) = \mathrm{id} - d(R),$

$\qquad r(R) = R^{\smile};R \cap \mathrm{id}, \qquad \mathrm{id} = \{(u,u) : u \in X\}$

Elements of $Rel(X)$ represent *actions* or *computations*

Operations model the *control flow* in the system

Multiplication represents e.g. *sequential* or *parallel composition* of actions

Addition (union) represents *nondeterministic choice*

Multiplicative units (id) model *ineffective actions* (*skip*)

Additive units ($\emptyset$) model *abortive actions*

E.g. *semigroups* or *monoids* model sequential composition

Semirings model sequential composition and nondeterministic choice

Concrete models of such algebras are sets of *partial* and *total functions*, *binary relations*, *languages*, *sets of computation paths* ...

Elements of $Rel(X)$ represent *actions* or *computations*

Operations model the *control flow* in the system

Multiplication represents e.g. *sequential* or *parallel composition* of actions

Addition (union) represents *nondeterministic choice*

Multiplicative units (id) model *ineffective actions* (*skip*)

Additive units ($\emptyset$) model *abortive actions*

E.g. *semigroups* or *monoids* model sequential composition

Semirings model sequential composition and nondeterministic choice

Concrete models of such algebras are sets of *partial* and *total functions*, *binary relations*, *languages*, *sets of computation paths* ...

Elements of $Rel(X)$ represent *actions* or *computations*

Operations model the *control flow* in the system

Multiplication represents e.g. *sequential* or *parallel composition* of actions

Addition (union) represents *nondeterministic choice*

Multiplicative units (id) model *ineffective actions* (*skip*)

Additive units ($\emptyset$) model *abortive actions*

E.g. *semigroups* or *monoids* model sequential composition

Semirings model sequential composition and nondeterministic choice

Concrete models of such algebras are sets of *partial* and *total functions*, *binary relations*, *languages*, *sets of computation paths* . . .

Elements of $Rel(X)$ represent *actions* or *computations*

Operations model the *control flow* in the system

Multiplication represents e.g. *sequential* or *parallel composition* of actions

Addition (union) represents *nondeterministic choice*

Multiplicative units (id) model *ineffective actions* (*skip*)

Additive units ($\emptyset$) model *abortive actions*

E.g. *semigroups* or *monoids* model sequential composition

Semirings model sequential composition and nondeterministic choice

Concrete models of such algebras are sets of *partial* and *total functions*, *binary relations*, *languages*, *sets of computation paths* ...

Elements of $Rel(X)$ represent *actions* or *computations*

Operations model the *control flow* in the system

Multiplication represents e.g. *sequential* or *parallel composition* of actions

Addition (union) represents *nondeterministic choice*

Multiplicative units (id) model *ineffective actions* (*skip*)

Additive units ($\emptyset$) model *abortive actions*

E.g. *semigroups* or *monoids* model sequential composition

Semirings model sequential composition and nondeterministic choice

Concrete models of such algebras are sets of *partial* and *total functions*, *binary relations*, *languages*, *sets of computation paths* ...

Elements of $Rel(X)$ represent *actions* or *computations*

Operations model the *control flow* in the system

Multiplication represents e.g. *sequential* or *parallel composition* of actions

Addition (union) represents *nondeterministic choice*

Multiplicative units (id) model *ineffective actions* (*skip*)

Additive units ($\emptyset$) model *abortive actions*

E.g. *semigroups* or *monoids* model sequential composition

Semirings model sequential composition and nondeterministic choice

Concrete models of such algebras are sets of *partial* and *total functions*, *binary relations*, *languages*, *sets of computation paths* ...

Elements of $Rel(X)$ represent *actions* or *computations*

Operations model the *control flow* in the system

Multiplication represents e.g. *sequential* or *parallel composition* of actions

Addition (union) represents *nondeterministic choice*

Multiplicative units (id) model *ineffective actions* (*skip*)

Additive units ($\emptyset$) model *abortive actions*

E.g. *semigroups* or *monoids* model sequential composition

Semirings model sequential composition and nondeterministic choice

Concrete models of such algebras are sets of *partial* and *total functions*, *binary relations*, *languages*, *sets of computation paths* ...

Elements of $Rel(X)$ represent *actions* or *computations*

Operations model the *control flow* in the system

Multiplication represents e.g. *sequential* or *parallel composition* of actions

Addition (union) represents *nondeterministic choice*

Multiplicative units (id) model *ineffective actions* (*skip*)

Additive units ($\emptyset$) model *abortive actions*

E.g. *semigroups* or *monoids* model sequential composition

Semirings model sequential composition and nondeterministic choice

Concrete models of such algebras are sets of *partial* and *total functions*, *binary relations*, *languages*, *sets of computation paths* . . .

Elements of $\mathit{Rel}(X)$ represent *actions* or *computations*

Operations model the *control flow* in the system

Multiplication represents e.g. *sequential* or *parallel composition* of actions

Addition (union) represents *nondeterministic choice*

Multiplicative units (id) model *ineffective actions* (*skip*)

Additive units ($\emptyset$) model *abortive actions*

E.g. *semigroups* or *monoids* model sequential composition

Semirings model sequential composition and nondeterministic choice

Concrete models of such algebras are sets of *partial* and *total functions*, *binary relations*, *languages*, *sets of computation paths* ...

A *domain* operation yields *enabledness conditions* for actions

I.e. the *domain* $d(x)$ of an action x models those states from which the action x *can be executed*

The *antidomain* $a(x)$ models those states from which the action x *cannot be executed*

Recall that a *semigroup* is a set with an associative binary operation $\cdot$

A *monoid* is a semigroup with an identity element 1

A *domain* operation yields *enabledness conditions* for actions

I.e. the *domain* $d(x)$ of an action x models those states from which the action x *can be executed*

The *antidomain* $a(x)$ models those states from which the action x *cannot be executed*

Recall that a *semigroup* is a set with an associative binary operation $\cdot$

A *monoid* is a semigroup with an identity element 1

A *domain* operation yields *enabledness conditions* for actions

I.e. the *domain* $d(x)$ of an action x models those states from which the action x *can be executed*

The *antidomain* $a(x)$ models those states from which the action x *cannot be executed*

Recall that a *semigroup* is a set with an associative binary operation $\cdot$

A *monoid* is a semigroup with an identity element 1

A *domain* operation yields *enabledness conditions* for actions

I.e. the *domain* $d(x)$ of an action x models those states from which the action x *can be executed*

The *antidomain* $a(x)$ models those states from which the action x *cannot be executed*

Recall that a *semigroup* is a set with an associative binary operation $\cdot$

A *monoid* is a semigroup with an identity element 1

A *domain semigroup*, or *d-semigroup*, is a semigroup $(S, \cdot)$ extended by a domain operation $d : S \to S$ that satisfies the following axioms

$$\begin{array}{ll} \text{(D1)} & d(x)x = x \\ \text{(D2)} & d(xy) = d(xd(y)) \\ \text{(D3)} & d(d(x)y) = d(x)d(y) \\ \text{(D4)} & d(x)d(y) = d(y)d(x) \end{array}$$

A monoid that satisfies these axioms is a *domain monoid* or *d-monoid*

It is easy to check that the axioms (D1)-(D4) hold in $Rel(X)$

(D2) is called the *locality axiom*

In semigroups, (D2) has also been called *left-congruence condition* [Jackson and Stokes 2001]

[Resende 2006] calls d *stable* if it satisfies (D2)

A *domain semigroup*, or *d-semigroup*, is a semigroup $(S, \cdot)$ extended by a domain operation $d : S \to S$ that satisfies the following axioms

$$
\begin{array}{ll}
\text{(D1)} & d(x)x = x \\
\text{(D2)} & d(xy) = d(xd(y)) \\
\text{(D3)} & d(d(x)y) = d(x)d(y) \\
\text{(D4)} & d(x)d(y) = d(y)d(x)
\end{array}
$$

A monoid that satisfies these axioms is a *domain monoid* or *d-monoid*

It is easy to check that the axioms (D1)-(D4) hold in $Rel(X)$

(D2) is called the *locality axiom*

In semigroups, (D2) has also been called *left-congruence condition* [Jackson and Stokes 2001]

[Resende 2006] calls d *stable* if it satisfies (D2)

A *domain semigroup*, or *d-semigroup*, is a semigroup $(S, \cdot)$ extended by a domain operation $d : S \to S$ that satisfies the following axioms

$$\begin{array}{ll} \text{(D1)} & d(x)x = x \\ \text{(D2)} & d(xy) = d(xd(y)) \\ \text{(D3)} & d(d(x)y) = d(x)d(y) \\ \text{(D4)} & d(x)d(y) = d(y)d(x) \end{array}$$

A monoid that satisfies these axioms is a *domain monoid* or *d-monoid*

It is easy to check that the axioms (D1)-(D4) hold in $Rel(X)$

(D2) is called the *locality axiom*

In semigroups, (D2) has also been called *left-congruence condition* [Jackson and Stokes 2001]

[Resende 2006] calls d *stable* if it satisfies (D2)

A *domain semigroup*, or *d-semigroup*, is a semigroup $(S, \cdot)$ extended by a domain operation $d : S \to S$ that satisfies the following axioms

$$
\begin{array}{ll}
\text{(D1)} & d(x)x = x \\
\text{(D2)} & d(xy) = d(xd(y)) \\
\text{(D3)} & d(d(x)y) = d(x)d(y) \\
\text{(D4)} & d(x)d(y) = d(y)d(x)
\end{array}
$$

A monoid that satisfies these axioms is a *domain monoid* or *d-monoid*

It is easy to check that the axioms (D1)-(D4) hold in $Rel(X)$

(D2) is called the *locality axiom*

In semigroups, (D2) has also been called *left-congruence condition* [Jackson and Stokes 2001]

[Resende 2006] calls d *stable* if it satisfies (D2)

A *domain semigroup*, or *d-semigroup*, is a semigroup $(S, \cdot)$ extended by a domain operation $d : S \to S$ that satisfies the following axioms

$$
\begin{array}{ll}
\text{(D1)} & d(x)x = x \\
\text{(D2)} & d(xy) = d(xd(y)) \\
\text{(D3)} & d(d(x)y) = d(x)d(y) \\
\text{(D4)} & d(x)d(y) = d(y)d(x)
\end{array}
$$

A monoid that satisfies these axioms is a *domain monoid* or *d-monoid*

It is easy to check that the axioms (D1)-(D4) hold in $Rel(X)$

(D2) is called the *locality axiom*

In semigroups, (D2) has also been called *left-congruence condition* [Jackson and Stokes 2001]

[Resende 2006] calls d *stable* if it satisfies (D2)

A *domain semigroup*, or *d-semigroup*, is a semigroup $(S, \cdot)$ extended by a domain operation $d : S \to S$ that satisfies the following axioms

$$\begin{array}{ll} \text{(D1)} & d(x)x = x \\ \text{(D2)} & d(xy) = d(xd(y)) \\ \text{(D3)} & d(d(x)y) = d(x)d(y) \\ \text{(D4)} & d(x)d(y) = d(y)d(x) \end{array}$$

A monoid that satisfies these axioms is a *domain monoid* or *d-monoid*

It is easy to check that the axioms (D1)-(D4) hold in $Rel(X)$

(D2) is called the *locality axiom*

In semigroups, (D2) has also been called *left-congruence condition* [Jackson and Stokes 2001]

[Resende 2006] calls d *stable* if it satisfies (D2)

A *domain semigroup*, or *d-semigroup*, is a semigroup $(S, \cdot)$ extended by a domain operation $d : S \to S$ that satisfies the following axioms

$$\begin{aligned} &\text{(D1)} \quad d(x)x = x \\ &\text{(D2)} \quad d(xy) = d(xd(y)) \\ &\text{(D3)} \quad d(d(x)y) = d(x)d(y) \\ &\text{(D4)} \quad d(x)d(y) = d(y)d(x) \end{aligned}$$

A monoid that satisfies these axioms is a *domain monoid* or *d-monoid*

It is easy to check that the axioms (D1)-(D4) hold in $Rel(X)$

(D2) is called the *locality axiom*

In semigroups, (D2) has also been called *left-congruence condition* [Jackson and Stokes 2001]

[Resende 2006] calls d *stable* if it satisfies (D2)

A *domain semigroup*, or *d-semigroup*, is a semigroup $(S, \cdot)$ extended by a domain operation $d : S \to S$ that satisfies the following axioms

$$\begin{array}{ll} \text{(D1)} & d(x)x = x \\ \text{(D2)} & d(xy) = d(xd(y)) \\ \text{(D3)} & d(d(x)y) = d(x)d(y) \\ \text{(D4)} & d(x)d(y) = d(y)d(x) \end{array}$$

A monoid that satisfies these axioms is a *domain monoid* or *d-monoid*

It is easy to check that the axioms (D1)-(D4) hold in $Rel(X)$

(D2) is called the *locality axiom*

In semigroups, (D2) has also been called *left-congruence condition* [Jackson and Stokes 2001]

[Resende 2006] calls d *stable* if it satisfies (D2)

A *domain semigroup*, or *d-semigroup*, is a semigroup $(S,\cdot)$ extended by a domain operation $d : S \to S$ that satisfies the following axioms

$$\begin{array}{ll}
\text{(D1)} & d(x)x = x \\
\text{(D2)} & d(xy) = d(xd(y)) \\
\text{(D3)} & d(d(x)y) = d(x)d(y) \\
\text{(D4)} & d(x)d(y) = d(y)d(x)
\end{array}$$

A monoid that satisfies these axioms is a *domain monoid* or *d-monoid*

It is easy to check that the axioms (D1)-(D4) hold in $Rel(X)$

(D2) is called the *locality axiom*

In semigroups, (D2) has also been called *left-congruence condition* [Jackson and Stokes 2001]

[Resende 2006] calls d *stable* if it satisfies (D2)

A *domain semigroup*, or *d-semigroup*, is a semigroup $(S, \cdot)$ extended by a domain operation $d : S \to S$ that satisfies the following axioms

$$\begin{array}{ll} \text{(D1)} & d(x)x = x \\ \text{(D2)} & d(xy) = d(xd(y)) \\ \text{(D3)} & d(d(x)y) = d(x)d(y) \\ \text{(D4)} & d(x)d(y) = d(y)d(x) \end{array}$$

A monoid that satisfies these axioms is a *domain monoid* or *d-monoid*

It is easy to check that the axioms (D1)-(D4) hold in $Rel(X)$

(D2) is called the *locality axiom*

In semigroups, (D2) has also been called *left-congruence condition* [Jackson and Stokes 2001]

[Resende 2006] calls d *stable* if it satisfies (D2)

A *quantale* $(Q, \bigvee, \cdot)$ is a complete join-semilattice $(Q, \bigvee)$

with an *associative* binary operation $\cdot$ on Q that

distributes over arbitrary joins in both variables.

$\bigvee \emptyset$ is denoted by 0, and $\bigvee Q$ is denoted by $\top$

Q is *unital* if $(Q, \cdot, e)$ is a *monoid* for some $e \in Q$

Q is *involutive* if there is a completely join-preserving *unary operation* $^\smile$ such that $x^{\smile\smile} = x$ and $(xy)^\smile = y^\smile x^\smile$

Quantale morphisms are maps that *preserve* arbitrary joins *and* $\cdot$

Morphisms of *unital* and/or *involutive* quantales also preserve e and/or $^\smile$

$\Longrightarrow$ 4 categories: $\mathbf{Qnt}$ $\mathbf{Qnt}_e$ $\mathbf{Qnt}_\smile$ $\mathbf{Qnt}_{e\smile}$

A *quantale* $(Q, \bigvee, \cdot)$ is a complete join-semilattice $(Q, \bigvee)$

with an *associative* binary operation $\cdot$ on Q that

distributes over arbitrary joins in both variables.

$\bigvee \emptyset$ is denoted by 0, and $\bigvee Q$ is denoted by $\top$

Q is *unital* if $(Q, \cdot, e)$ is a *monoid* for some $e \in Q$

Q is *involutive* if there is a completely join-preserving *unary operation* $^{\smile}$ such that $x^{\smile\smile} = x$ and $(xy)^{\smile} = y^{\smile}x^{\smile}$

Quantale morphisms are maps that *preserve* arbitrary joins *and* $\cdot$

Morphisms of *unital* and/or *involutive* quantales also preserve e and/or $^{\smile}$

$\implies$ 4 categories: $\mathbf{Qnt}$ $\mathbf{Qnt}_e$ $\mathbf{Qnt}_{\smile}$ $\mathbf{Qnt}_{e\smile}$

A *quantale* $(Q, \bigvee, \cdot)$ is a complete join-semilattice $(Q, \bigvee)$

with an *associative* binary operation $\cdot$ on Q that

distributes over arbitrary joins in both variables.

$\bigvee \emptyset$ is denoted by 0, and $\bigvee Q$ is denoted by $\top$

Q is *unital* if $(Q, \cdot, e)$ is a *monoid* for some $e \in Q$

Q is *involutive* if there is a completely join-preserving *unary operation* $^{\smile}$ such that $x^{\smile\smile} = x$ and $(xy)^{\smile} = y^{\smile}x^{\smile}$

Quantale morphisms are maps that *preserve* arbitrary joins *and* $\cdot$

Morphisms of *unital* and/or *involutive* quantales also preserve e and/or $^{\smile}$

$\Longrightarrow$ 4 categories: **Qnt** $\quad$ $\mathbf{Qnt}_e$ $\quad$ $\mathbf{Qnt}_{\smile}$ $\quad$ $\mathbf{Qnt}_{e\smile}$

A *quantale* $(Q, \bigvee, \cdot)$ is a complete join-semilattice $(Q, \bigvee)$

with an *associative* binary operation $\cdot$ on Q that

distributes over arbitrary joins in both variables.

$\bigvee \emptyset$ is denoted by 0, and $\bigvee Q$ is denoted by $\top$

Q is *unital* if $(Q, \cdot, e)$ is a *monoid* for some $e \in Q$

Q is *involutive* if there is a completely join-preserving *unary operation* $^\smile$ such that $x^{\smile\smile} = x$ and $(xy)^\smile = y^\smile x^\smile$

Quantale morphisms are maps that *preserve* arbitrary joins *and* $\cdot$

Morphisms of *unital* and/or *involutive* quantales also preserve *e* and/or $^\smile$

$\Longrightarrow$ 4 categories: $\mathbf{Qnt}$ $\mathbf{Qnt}_e$ $\mathbf{Qnt}_\smile$ $\mathbf{Qnt}_{e\smile}$

A *quantale* $(Q, \bigvee, \cdot)$ is a complete join-semilattice $(Q, \bigvee)$

with an *associative* binary operation $\cdot$ on Q that

distributes over arbitrary joins in both variables.

$\bigvee \emptyset$ is denoted by 0, and $\bigvee Q$ is denoted by $\top$

Q is *unital* if $(Q, \cdot, e)$ is a *monoid* for some $e \in Q$

Q is *involutive* if there is a completely join-preserving *unary operation* $^\smile$ such that $x^{\smile\smile} = x$ and $(xy)^\smile = y^\smile x^\smile$

Quantale morphisms are maps that *preserve* arbitrary joins *and* $\cdot$

Morphisms of *unital* and/or *involutive* quantales also preserve e and/or $^\smile$

$\Longrightarrow$ 4 categories: $\mathbf{Qnt}$ $\mathbf{Qnt}_e$ $\mathbf{Qnt}_\smile$ $\mathbf{Qnt}_{e\smile}$

A *quantale* $(Q, \bigvee, \cdot)$ is a complete join-semilattice $(Q, \bigvee)$

with an *associative* binary operation $\cdot$ on Q that

distributes over arbitrary joins in both variables.

$\bigvee \emptyset$ is denoted by 0, and $\bigvee Q$ is denoted by $\top$

Q is *unital* if $(Q, \cdot, e)$ is a *monoid* for some $e \in Q$

Q is *involutive* if there is a completely join-preserving *unary operation* $^{\smile}$ such that $x^{\smile\smile} = x$ and $(xy)^{\smile} = y^{\smile}x^{\smile}$

Quantale morphisms are maps that *preserve* arbitrary joins *and* $\cdot$

Morphisms of *unital* and/or *involutive* quantales also preserve e and/or $^{\smile}$

$\Longrightarrow$ 4 categories: **Qnt** $\quad$ **Qnt**$_e$ $\quad$ **Qnt**$_{\smile}$ $\quad$ **Qnt**$_{e\smile}$

A *quantale* $(Q, \bigvee, \cdot)$ is a complete join-semilattice $(Q, \bigvee)$

with an *associative* binary operation $\cdot$ on Q that

distributes over arbitrary joins in both variables.

$\bigvee \emptyset$ is denoted by 0, and $\bigvee Q$ is denoted by $\top$

Q is *unital* if $(Q, \cdot, e)$ is a *monoid* for some $e \in Q$

Q is *involutive* if there is a completely join-preserving *unary operation* $^\smile$ such that $x^{\smile\smile} = x$ and $(xy)^\smile = y^\smile x^\smile$

Quantale morphisms are maps that *preserve* arbitrary joins *and* $\cdot$

Morphisms of *unital* and/or *involutive* quantales also preserve e and/or $^\smile$

$\Longrightarrow$ 4 categories: $\mathbf{Qnt}$ $\quad \mathbf{Qnt}_e$ $\quad \mathbf{Qnt}_\smile$ $\quad \mathbf{Qnt}_{e\smile}$

Examples

For any *semigroup* $(S, \cdot)$ the complex algebra $(\mathcal{P}(S), \bigcup, \cdot)$ is a quantale, where

$$X \cdot Y = \{xy : x \in X, y \in Y\}$$

If $(S, \cdot, e)$ is a *monoid* then $(\mathcal{P}(S), \bigcup, \cdot, \{e\})$ is a unital quantale

If $(S, \cdot, ^{-1}, e)$ is a *group* then $(\mathcal{P}(S), \bigcup, \cdot, ^{-1}, \{e\})$ is a unital involutive quantale

For any set U the *full relation algebra*

$$Rel(U) = (\mathcal{P}(U^2), \bigcup, \bigcap, -, ;, ^{\smile}, \mathrm{id}_U)$$

has a *unital involutive quantale reduct*

Any complete residuated lattice has a *quantale reduct* if one forgets $\wedge, \backslash, /$

Examples

For any *semigroup* $(S, \cdot)$ the complex algebra $(\mathcal{P}(S), \bigcup, \cdot)$ is a quantale, where

$$X \cdot Y = \{xy : x \in X, y \in Y\}$$

If $(S, \cdot, e)$ is a *monoid* then $(\mathcal{P}(S), \bigcup, \cdot, \{e\})$ is a unital quantale

If $(S, \cdot, ^{-1}, e)$ is a *group* then $(\mathcal{P}(S), \bigcup, \cdot, ^{-1}, \{e\})$ is a unital involutive quantale

For any set U the *full relation algebra*

$$Rel(U) = (\mathcal{P}(U^2), \bigcup, \bigcap, -, ;, ^{\smile}, \mathrm{id}_U)$$

has a *unital involutive quantale reduct*

Any complete residuated lattice has a *quantale reduct* if one forgets $\wedge, \backslash, /$

Examples

For any *semigroup* $(S,\cdot)$ the complex algebra $(\mathcal{P}(S),\bigcup,\cdot)$ is a quantale, where

$$X \cdot Y = \{xy : x \in X, y \in Y\}$$

If $(S,\cdot,e)$ is a *monoid* then $(\mathcal{P}(S),\bigcup,\cdot,\{e\})$ is a unital quantale

If $(S,\cdot,^{-1},e)$ is a *group* then $(\mathcal{P}(S),\bigcup,\cdot,^{-1},\{e\})$ is a unital involutive quantale

For any set U the *full relation algebra*

$$Rel(U) = (\mathcal{P}(U^2),\bigcup,\bigcap,-,;,^{\smile},\mathrm{id}_U)$$

has a *unital involutive quantale reduct*

Any complete residuated lattice has a *quantale reduct* if one forgets $\wedge, \backslash, /$

Examples

For any *semigroup* $(S, \cdot)$ the complex algebra $(\mathcal{P}(S), \bigcup, \cdot)$ is a quantale, where

$$X \cdot Y = \{xy : x \in X, y \in Y\}$$

If $(S, \cdot, e)$ is a *monoid* then $(\mathcal{P}(S), \bigcup, \cdot, \{e\})$ is a unital quantale

If $(S, \cdot, ^{-1}, e)$ is a *group* then $(\mathcal{P}(S), \bigcup, \cdot, ^{-1}, \{e\})$ is a unital involutive quantale

For any set U the *full relation algebra*

$$Rel(U) = (\mathcal{P}(U^2), \bigcup, \bigcap, -, ;, ^{\smile}, \mathrm{id}_U)$$

has a *unital involutive quantale reduct*

Any complete residuated lattice has a *quantale reduct* if one forgets $\wedge, \backslash, /$

Examples

For any *semigroup* $(S, \cdot)$ the complex algebra $(\mathcal{P}(S), \bigcup, \cdot)$ is a quantale, where

$$X \cdot Y = \{xy : x \in X, y \in Y\}$$

If $(S, \cdot, e)$ is a *monoid* then $(\mathcal{P}(S), \bigcup, \cdot, \{e\})$ is a unital quantale

If $(S, \cdot, ^{-1}, e)$ is a *group* then $(\mathcal{P}(S), \bigcup, \cdot, ^{-1}, \{e\})$ is a unital involutive quantale

For any set U the *full relation algebra*

$$Rel(U) = (\mathcal{P}(U^2), \bigcup, \bigcap, -, ;, ^{\smile}, \mathrm{id}_U)$$

has a *unital involutive quantale reduct*

Any complete residuated lattice has a *quantale reduct* if one forgets $\wedge, \backslash, /$

Any *frame* is a unital involutive quantale if we define $xy = x \wedge y$, $x^{\smile} = x$

The category **Frm** of frames is a full subcategory of $\mathbf{Qnt}_e$ and $\mathbf{Qnt}_{e^{\smile}}$

A *stably supported quantale* $(Q, \bigvee, \cdot, ^{\smile}, d, e)$ is a unital involutive quantale with a completely join-preserving $\varsigma : Q \to Q$ that satisfies

$$\varsigma(x) \le e \qquad \varsigma(x) \le xx^{\smile}$$
$$\varsigma(x)x = x$$
$$\varsigma(xy) = \varsigma(x\varsigma(y))$$
$$\varsigma(\varsigma(x)y) = \varsigma(x)\varsigma(y)$$
$$\varsigma(x)\varsigma(y) = \varsigma(y)\varsigma(x)$$

The last 4 are the same axioms as for domain semigroups!

ς is called the *support* of Q, and (D2) makes it *stable*

Will denote the support by d rather than ς

Any *frame* is a unital involutive quantale if we define $xy = x \wedge y$, $x^{\smile} = x$

The category **Frm** of frames is a full subcategory of $\mathbf{Qnt}_e$ and $\mathbf{Qnt}_{e^{\smile}}$

A *stably supported quantale* $(Q, \bigvee, \cdot, ^{\smile}, d, e)$ is a unital involutive quantale with a completely join-preserving $\varsigma : Q \to Q$ that satisfies

$$\varsigma(x) \le e \qquad \varsigma(x) \le xx^{\smile}$$
$$\varsigma(x)x = x$$
$$\varsigma(xy) = \varsigma(x\varsigma(y))$$
$$\varsigma(\varsigma(x)y) = \varsigma(x)\varsigma(y)$$
$$\varsigma(x)\varsigma(y) = \varsigma(y)\varsigma(x)$$

The last 4 are the same axioms as for domain semigroups!

ς is called the *support* of Q, and (D2) makes it *stable*

Will denote the support by d rather than ς

Any *frame* is a unital involutive quantale if we define $xy = x \wedge y$, $x^{\smile} = x$

The category **Frm** of frames is a full subcategory of $\mathbf{Qnt}_e$ and $\mathbf{Qnt}_{e^{\smile}}$

A *stably supported quantale* $(Q, \bigvee, \cdot, ^{\smile}, d, e)$ is a unital involutive quantale with a completely join-preserving $\varsigma : Q \to Q$ that satisfies

$$\varsigma(x) \leq e \qquad \varsigma(x) \leq xx^{\smile}$$
$$\varsigma(x)x = x$$
$$\varsigma(xy) = \varsigma(x\varsigma(y))$$
$$\varsigma(\varsigma(x)y) = \varsigma(x)\varsigma(y)$$
$$\varsigma(x)\varsigma(y) = \varsigma(y)\varsigma(x)$$

The last 4 are the same axioms as for domain semigroups!

ς is called the *support* of Q, and (D2) makes it *stable*

Will denote the support by d rather than ς

Any *frame* is a unital involutive quantale if we define $xy = x \wedge y$, $x^{\smile} = x$

The category **Frm** of frames is a full subcategory of $\mathbf{Qnt}_e$ and $\mathbf{Qnt}_{e^{\smile}}$

A *stably supported quantale* $(Q, \bigvee, \cdot, ^{\smile}, d, e)$ is a unital involutive quantale with a completely join-preserving $\varsigma : Q \to Q$ that satisfies

$$\varsigma(x) \le e \qquad \varsigma(x) \le xx^{\smile}$$

$$\varsigma(x)x = x$$

$$\varsigma(xy) = \varsigma(x\varsigma(y))$$

$$\varsigma(\varsigma(x)y) = \varsigma(x)\varsigma(y)$$

$$\varsigma(x)\varsigma(y) = \varsigma(y)\varsigma(x)$$

The last 4 are the same axioms as for domain semigroups!

ς is called the *support* of Q, and (D2) makes it *stable*

Will denote the support by d rather than ς

Any *frame* is a unital involutive quantale if we define $xy = x \wedge y$, $x^{\smile} = x$

The category **Frm** of frames is a full subcategory of $\mathbf{Qnt}_e$ and $\mathbf{Qnt}_{e\smile}$

A *stably supported quantale* $(Q, \bigvee, \cdot, ^{\smile}, d, e)$ is a unital involutive quantale with a completely join-preserving $\varsigma : Q \to Q$ that satisfies

$$\varsigma(x) \leq e \qquad \varsigma(x) \leq xx^{\smile}$$
$$\varsigma(x)x = x$$
$$\varsigma(xy) = \varsigma(x\varsigma(y))$$
$$\varsigma(\varsigma(x)y) = \varsigma(x)\varsigma(y)$$
$$\varsigma(x)\varsigma(y) = \varsigma(y)\varsigma(x)$$

The last 4 are the same axioms as for domain semigroups!

ς is called the *support* of Q, and (D2) makes it *stable*

Will denote the support by d rather than ς

Any *frame* is a unital involutive quantale if we define $xy = x \wedge y$, $x^{\smile} = x$

The category **Frm** of frames is a full subcategory of $\mathbf{Qnt}_e$ and $\mathbf{Qnt}_{e\smile}$

A *stably supported quantale* $(Q, \bigvee, \cdot, ^{\smile}, d, e)$ is a unital involutive quantale with a completely join-preserving $\varsigma : Q \to Q$ that satisfies

$$\varsigma(x) \le e \qquad \varsigma(x) \le xx^{\smile}$$
$$\varsigma(x)x = x$$
$$\varsigma(xy) = \varsigma(x\varsigma(y))$$
$$\varsigma(\varsigma(x)y) = \varsigma(x)\varsigma(y)$$
$$\varsigma(x)\varsigma(y) = \varsigma(y)\varsigma(x)$$

The last 4 are the same axioms as for domain semigroups!

ς is called the *support* of Q, and (D2) makes it *stable*

Will denote the support by d rather than ς

Any *frame* is a unital involutive quantale if we define $xy = x \wedge y$, $x^{\smile} = x$

The category **Frm** of frames is a full subcategory of $\mathbf{Qnt}_e$ and $\mathbf{Qnt}_{e\smile}$

A *stably supported quantale* $(Q, \bigvee, \cdot, ^{\smile}, d, e)$ is a unital involutive quantale with a completely join-preserving $\varsigma : Q \to Q$ that satisfies

$$\varsigma(x) \leq e \qquad \varsigma(x) \leq xx^{\smile}$$
$$\varsigma(x)x = x$$
$$\varsigma(xy) = \varsigma(x\varsigma(y))$$
$$\varsigma(\varsigma(x)y) = \varsigma(x)\varsigma(y)$$
$$\varsigma(x)\varsigma(y) = \varsigma(y)\varsigma(x)$$

The last 4 are the same axioms as for domain semigroups!

ς is called the *support* of Q, and (D2) makes it *stable*

Will denote the support by d rather than ς

Any *frame* is a unital involutive quantale if we define $xy = x \wedge y$, $x^{\smile} = x$

The category **Frm** of frames is a full subcategory of $\mathbf{Qnt}_e$ and $\mathbf{Qnt}_{e\smile}$

A *stably supported quantale* $(Q, \bigvee, \cdot, ^{\smile}, d, e)$ is a unital involutive quantale with a completely join-preserving $\varsigma : Q \to Q$ that satisfies

$$\varsigma(x) \leq e \qquad \varsigma(x) \leq xx^{\smile}$$
$$\varsigma(x)x = x$$
$$\varsigma(xy) = \varsigma(x\varsigma(y))$$
$$\varsigma(\varsigma(x)y) = \varsigma(x)\varsigma(y)$$
$$\varsigma(x)\varsigma(y) = \varsigma(y)\varsigma(x)$$

The last 4 are the same axioms as for domain semigroups!

ς is called the *support* of Q, and (D2) makes it *stable*

Will denote the support by d rather than ς

Any *frame* is a unital involutive quantale if we define $xy = x \wedge y$, $x^{\smile} = x$

The category **Frm** of frames is a full subcategory of $\mathbf{Qnt}_e$ and $\mathbf{Qnt}_{e\smile}$

A *stably supported quantale* $(Q, \bigvee, \cdot, ^{\smile}, d, e)$ is a unital involutive quantale with a completely join-preserving $\varsigma : Q \to Q$ that satisfies

$$\begin{aligned}
&\varsigma(x) \le e \qquad \varsigma(x) \le xx^{\smile} \\
&\varsigma(x)x = x \\
&\varsigma(xy) = \varsigma(x\varsigma(y)) \\
&\varsigma(\varsigma(x)y) = \varsigma(x)\varsigma(y) \\
&\varsigma(x)\varsigma(y) = \varsigma(y)\varsigma(x)
\end{aligned}$$

The last 4 are the same axioms as for domain semigroups!

ς is called the *support* of Q, and (D2) makes it *stable*

Will denote the support by d rather than ς

Any *frame* is a unital involutive quantale if we define $xy = x \wedge y$, $x^{\smile} = x$

The category **Frm** of frames is a full subcategory of $\mathbf{Qnt}_e$ and $\mathbf{Qnt}_{e\smile}$

A *stably supported quantale* $(Q, \bigvee, \cdot, ^{\smile}, d, e)$ is a unital involutive quantale with a completely join-preserving $\varsigma : Q \to Q$ that satisfies

$$\begin{aligned}
&\varsigma(x) \leq e \qquad \varsigma(x) \leq xx^{\smile} \\
&\varsigma(x)x = x \\
&\varsigma(xy) = \varsigma(x\varsigma(y)) \\
&\varsigma(\varsigma(x)y) = \varsigma(x)\varsigma(y) \\
&\varsigma(x)\varsigma(y) = \varsigma(y)\varsigma(x)
\end{aligned}$$

The last 4 are the same axioms as for domain semigroups!

ς is called the *support* of Q, and (D2) makes it *stable*

Will denote the support by d rather than ς

Any *frame* is a unital involutive quantale if we define $xy = x \wedge y$, $x^{\smile} = x$

The category **Frm** of frames is a full subcategory of $\mathbf{Qnt}_e$ and $\mathbf{Qnt}_{e\smile}$

A *stably supported quantale* $(Q, \bigvee, \cdot, ^{\smile}, d, e)$ is a unital involutive quantale with a completely join-preserving $\varsigma : Q \to Q$ that satisfies

$$\begin{aligned}
&\varsigma(x) \leq e \qquad \varsigma(x) \leq xx^{\smile} \\
&\varsigma(x)x = x \\
&\varsigma(xy) = \varsigma(x\varsigma(y)) \\
&\varsigma(\varsigma(x)y) = \varsigma(x)\varsigma(y) \\
&\varsigma(x)\varsigma(y) = \varsigma(y)\varsigma(x)
\end{aligned}$$

The last 4 are the same axioms as for domain semigroups!

ς is called the *support* of Q, and (D2) makes it *stable*

Will denote the support by d rather than ς

From the involution one can define $r(x) = d(x^{\smile})$

However in CS applications, $^{\smile}$ is not so prominent

A *domain quantale* $(Q, \bigvee, \cdot, d, e)$ is a unital quantale with a domain operation d that is completely join-preserving and satisfies $d(x) \vee e = e$

So *domain quantales* are stably supported quantales without $^{\smile}$ axioms

Lemma

If Q is a domain quantale then $d(Q) = \{d(x) : x \in Q\}$ is a frame with $x \wedge y = xy$

If Q is a stably supported quantale then $d(Q) = \downarrow e$

From the involution one can define $r(x) = d(x^{\smile})$

However in CS applications, $^{\smile}$ is not so prominent

A *domain quantale* $(Q, \bigvee, \cdot, d, e)$ is a unital quantale with a domain operation d that is completely join-preserving and satisfies $d(x) \vee e = e$

So *domain quantales* are stably supported quantales without $^{\smile}$ axioms

Lemma

If Q is a domain quantale then $d(Q) = \{d(x) : x \in Q\}$ is a frame with $x \wedge y = xy$

If Q is a stably supported quantale then $d(Q) = \downarrow e$

From the involution one can define $r(x) = d(x^{\smile})$

However in CS applications, $^{\smile}$ is not so prominent

A *domain quantale* $(Q, \bigvee, \cdot, d, e)$ is a unital quantale with a domain operation d that is completely join-preserving and satisfies $d(x) \vee e = e$

So *domain quantales* are stably supported quantales without $^{\smile}$ axioms

Lemma

If Q is a domain quantale then $d(Q) = \{d(x) : x \in Q\}$ is a frame with $x \wedge y = xy$

If Q is a stably supported quantale then $d(Q) = \downarrow e$

From the involution one can define $r(x) = d(x^{\smile})$

However in CS applications, $^{\smile}$ is not so prominent

A *domain quantale* $(Q, \bigvee, \cdot, d, e)$ is a unital quantale with a domain operation d that is completely join-preserving and satisfies $d(x) \vee e = e$

So *domain quantales* are stably supported quantales without $^{\smile}$ axioms

Lemma

If Q is a domain quantale then $d(Q) = \{d(x) : x \in Q\}$ is a frame with $x \wedge y = xy$

If Q is a stably supported quantale then $d(Q) = \downarrow e$

From the involution one can define $r(x) = d(x^{\smile})$

However in CS applications, $^{\smile}$ is not so prominent

A *domain quantale* $(Q, \bigvee, \cdot, d, e)$ is a unital quantale with a domain operation d that is completely join-preserving and satisfies $d(x) \vee e = e$

So *domain quantales* are stably supported quantales without $^{\smile}$ axioms

Lemma

If Q is a domain quantale then $d(Q) = \{d(x) : x \in Q\}$ is a frame with $x \wedge y = xy$

If Q is a stably supported quantale then $d(Q) = \downarrow e$

From the involution one can define $r(x) = d(x^{\smile})$

However in CS applications, $^{\smile}$ is not so prominent

A *domain quantale* $(Q, \bigvee, \cdot, d, e)$ is a unital quantale with a domain operation d that is completely join-preserving and satisfies $d(x) \vee e = e$

So *domain quantales* are stably supported quantales without $^{\smile}$ axioms

Lemma

If Q is a domain quantale then $d(Q) = \{d(x) : x \in Q\}$ is a frame with $x \wedge y = xy$

If Q is a stably supported quantale then $d(Q) = \downarrow e$

Program semantics from domain quantales

The complemented elements in the frame $d(Q)$ are considered as Boolean predicates $p = d(x)$ with complement $\bar{p}$

Can be used for modeling the programming constructs "if p then x else y" by $px \vee \bar{p}y$

"while p do x" by $(\bigvee_{n=0}^{<\omega}(px)^n)\bar{p}$,

where $z^0 = e$ and $z^{n+1} = zz^n$.

Hence in this model of program semantics the *frame* contains predicates over a state space and

the *quantale* contains programs that transform predicates by multiplication.

Program semantics from domain quantales

The complemented elements in the frame $d(Q)$ are considered as Boolean predicates $p = d(x)$ with complement $\bar{p}$

Can be used for modeling the programming constructs "if p then x else y" by $px \vee \bar{p}y$

"while p do x" by $(\bigvee_{n=0}^{<\omega}(px)^n)\bar{p}$,

where $z^0 = e$ and $z^{n+1} = zz^n$.

Hence in this model of program semantics the *frame* contains predicates over a state space and

the *quantale* contains programs that transform predicates by multiplication.

Program semantics from domain quantales

The complemented elements in the frame $d(Q)$ are considered as Boolean predicates $p = d(x)$ with complement $\bar{p}$

Can be used for modeling the programming constructs "if p then x else y" by $px \vee \bar{p}y$

"while p do x" by $(\bigvee_{n=0}^{<\omega}(px)^n)\bar{p}$,

where $z^0 = e$ and $z^{n+1} = zz^n$.

Hence in this model of program semantics the *frame* contains predicates over a state space and

the *quantale* contains programs that transform predicates by multiplication.

Program semantics from domain quantales

The complemented elements in the frame $d(Q)$ are considered as Boolean predicates $p = d(x)$ with complement $\bar{p}$

Can be used for modeling the programming constructs "if p then x else y" by $px \vee \bar{p}y$

"while p do x" by $(\bigvee_{n=0}^{<\omega}(px)^n)\bar{p}$,

where $z^0 = e$ and $z^{n+1} = zz^n$.

Hence in this model of program semantics the *frame* contains predicates over a state space and

the *quantale* contains programs that transform predicates by multiplication.

Program semantics from domain quantales

The complemented elements in the frame $d(Q)$ are considered as Boolean predicates $p = d(x)$ with complement $\bar{p}$

Can be used for modeling the programming constructs "if p then x else y" by $px \vee \bar{p}y$

"while p do x" by $(\bigvee_{n=0}^{<\omega}(px)^n)\bar{p}$,

where $z^0 = e$ and $z^{n+1} = zz^n$.

Hence in this model of program semantics the *frame* contains predicates over a state space and

the *quantale* contains programs that transform predicates by multiplication.

Domain quantales from domain monoids

Domain monoids produce domain quantales by a modified powerset construction:

Let $(S, \cdot, d, e)$ be a domain monoid and define the *fundamental order* on S by

$$x \leq y \;\Leftrightarrow\; x = d(x)y$$

and let $\mathcal{D}(S)$ be all downward closed subsets of S

Then $(\mathcal{D}(S), \bigcup, \cdot, d, {\downarrow}e)$ is a domain quantale where for $X, Y \in D(S)$

$$X \cdot Y = {\downarrow}\{xy : x \in X, y \in Y\} \qquad d(X) = {\downarrow}\{d(x) : x \in X\}$$

Another example of a domain quantale is $Rel(U)$ with $\bigcup$, composition, and domain

Domain quantales from domain monoids

Domain monoids produce domain quantales by a modified powerset construction:

Let $(S, \cdot, d, e)$ be a domain monoid and define the *fundamental order* on S by

$$x \leq y \;\Leftrightarrow\; x = d(x)y$$

and let $\mathcal{D}(S)$ be all downward closed subsets of S

Then $(\mathcal{D}(S), \bigcup, \cdot, d, {\downarrow}e)$ is a domain quantale where for $X, Y \in D(S)$

$$X \cdot Y = {\downarrow}\{xy : x \in X, y \in Y\} \qquad d(X) = {\downarrow}\{d(x) : x \in X\}$$

Another example of a domain quantale is $Rel(U)$ with $\bigcup$, composition, and domain

Domain quantales from domain monoids

Domain monoids produce domain quantales by a modified powerset construction:

Let $(S, \cdot, d, e)$ be a domain monoid and define the *fundamental order* on S by

$$x \leq y \;\Leftrightarrow\; x = d(x)y$$

and let $\mathcal{D}(S)$ be all downward closed subsets of S

Then $(\mathcal{D}(S), \bigcup, \cdot, d, \downarrow e)$ is a domain quantale where for $X, Y \in D(S)$

$$X \cdot Y = \downarrow\{xy : x \in X, y \in Y\} \qquad d(X) = \downarrow\{d(x) : x \in X\}$$

Another example of a domain quantale is $Rel(U)$ with $\bigcup$, composition, and domain

Representable d-monoids

Tarski [1948] defined the class RA of abstract relation algebras and asked if every relation algebra is *representable*, i.e. embeddable into an algebra of binary relations

Monk [1964] proved that the class RRA of representable relation algebras is *not finitely axiomatizable*

Does the axiomatisation of d-monoids captures all the properties of the domain operation of binary relations?

A d-monoid is called *representable* if it can be embedded in $Rel(X)$ for some set X such that $\cdot$, d and 1 correspond to composition, relational domain and id_X

By the fundamental theorem for relation algebras [Schein 1970] the class of representable d-monoids is a *quasivariety*

Representable d-monoids

Tarski [1948] defined the class RA of abstract relation algebras and asked if every relation algebra is *representable*, i.e. embeddable into an algebra of binary relations

Monk [1964] proved that the class RRA of representable relation algebras is *not finitely axiomatizable*

Does the axiomatisation of d-monoids captures all the properties of the domain operation of binary relations?

A d-monoid is called *representable* if it can be embedded in $Rel(X)$ for some set X such that $\cdot$, d and 1 correspond to composition, relational domain and id_X

By the fundamental theorem for relation algebras [Schein 1970] the class of representable d-monoids is a *quasivariety*

Representable d-monoids

Tarski [1948] defined the class RA of abstract relation algebras and asked if every relation algebra is *representable*, i.e. embeddable into an algebra of binary relations

Monk [1964] proved that the class RRA of representable relation algebras is *not finitely axiomatizable*

Does the axiomatisation of d-monoids captures all the properties of the domain operation of binary relations?

A d-monoid is called *representable* if it can be embedded in $Rel(X)$ for some set X such that $\cdot$, d and 1 correspond to composition, relational domain and id_X

By the fundamental theorem for relation algebras [Schein 1970] the class of representable d-monoids is a *quasivariety*

Representable d-monoids

Tarski [1948] defined the class RA of abstract relation algebras and asked if every relation algebra is *representable*, i.e. embeddable into an algebra of binary relations

Monk [1964] proved that the class RRA of representable relation algebras is *not finitely axiomatizable*

Does the axiomatisation of d-monoids captures all the properties of the domain operation of binary relations?

A d-monoid is called *representable* if it can be embedded in $Rel(X)$ for some set X such that $\cdot$, d and 1 correspond to composition, relational domain and id_X

By the fundamental theorem for relation algebras [Schein 1970] the class of representable d-monoids is a *quasivariety*

Representable *d*-monoids

Tarski [1948] defined the class RA of abstract relation algebras and asked if every relation algebra is *representable*, i.e. embeddable into an algebra of binary relations

Monk [1964] proved that the class RRA of representable relation algebras is *not finitely axiomatizable*

Does the axiomatisation of d-monoids captures all the properties of the domain operation of binary relations?

A d-monoid is called *representable* if it can be embedded in $Rel(X)$ for some set X such that $\cdot$, d and 1 correspond to composition, relational domain and id_X

By the fundamental theorem for relation algebras [Schein 1970] the class of representable d-monoids is a *quasivariety*

Proposition

The following quasiequation fails in a 4-element d-monoid but holds in Rel(X): $xy = d(x)$ *and* $yx = x$ *and* $d(y) = 1$ *imply* $x = d(x)$

Proof.

Finding a 4-element counterexample for d-monoids is straight forward

To prove the result for $Rel(X)$, consider $x, y \in Rel(X)$ and $(a, b) \in x$.

Then $d(y) = 1$ implies $(b, c) \in y$ for some c.

It follows from $xy = d(x)$ that $c = a$, hence $(b, a) \in y$.

Now $yx = x$ implies that $(b, b) \in x$.

Finally $xy = d(x)$ yields $(b, a) \in d(x)$, whence $b = a$.

Since (a, b) is arbitrary it follows that $x = d(x)$. □

Corollary

The quasivariety of representable d-monoids is not a variety

Proposition

The following quasiequation fails in a 4-element d-monoid but holds in $Rel(X)$*:* $xy = d(x)$ *and* $yx = x$ *and* $d(y) = 1$ *imply* $x = d(x)$

Proof.

Finding a 4-element counterexample for d-monoids is straight forward

To prove the result for $Rel(X)$, consider $x, y \in Rel(X)$ and $(a, b) \in x$.

Then $d(y) = 1$ implies $(b, c) \in y$ for some c.

It follows from $xy = d(x)$ that $c = a$, hence $(b, a) \in y$.

Now $yx = x$ implies that $(b, b) \in x$.

Finally $xy = d(x)$ yields $(b, a) \in d(x)$, whence $b = a$.

Since (a, b) is arbitrary it follows that $x = d(x)$. □

Corollary

The quasivariety of representable d-monoids is not a variety

Proposition

The following quasiequation fails in a 4-element d-monoid but holds in $Rel(X)$*:* $xy = d(x)$ *and* $yx = x$ *and* $d(y) = 1$ *imply* $x = d(x)$

Proof.

Finding a 4-element counterexample for d-monoids is straight forward

To prove the result for $Rel(X)$, consider $x, y \in Rel(X)$ and $(a, b) \in x$.

Then $d(y) = 1$ implies $(b, c) \in y$ for some c.

It follows from $xy = d(x)$ that $c = a$, hence $(b, a) \in y$.

Now $yx = x$ implies that $(b, b) \in x$.

Finally $xy = d(x)$ yields $(b, a) \in d(x)$, whence $b = a$.

Since (a, b) is arbitrary it follows that $x = d(x)$. □

Corollary

The quasivariety of representable d-monoids is not a variety

Twisted Domain Semigroups

Partial functions under composition satisfy another equational property called the *twisted law* by [Jackson and Stokes 2001]:

$$xd(y) = d(xy)x$$

This identity fails in $Rel(X)$ if we take x to be any relation that is not *deterministic*

A d-semigroup/monoid is *twisted* if it satisfies the twisted law

[Manes 2006] refers to them as *guarded* semigroups/monoids

$d(x)$ is the *guard* of x

Twisted Domain Semigroups

Partial functions under composition satisfy another equational property called the *twisted law* by [Jackson and Stokes 2001]:

$$xd(y) = d(xy)x$$

This identity fails in $Rel(X)$ if we take x to be any relation that is not *deterministic*

A d-semigroup/monoid is *twisted* if it satisfies the twisted law

[Manes 2006] refers to them as *guarded* semigroups/monoids

$d(x)$ is the *guard* of x

Twisted Domain Semigroups

Partial functions under composition satisfy another equational property called the *twisted law* by [Jackson and Stokes 2001]:

$$xd(y) = d(xy)x$$

This identity fails in $Rel(X)$ if we take x to be any relation that is not *deterministic*

A d-semigroup/monoid is *twisted* if it satisfies the twisted law

[Manes 2006] refers to them as *guarded* semigroups/monoids

$d(x)$ is the *guard* of x

Twisted Domain Semigroups

Partial functions under composition satisfy another equational property called the *twisted law* by [Jackson and Stokes 2001]:

$$xd(y) = d(xy)x$$

This identity fails in $Rel(X)$ if we take x to be any relation that is not *deterministic*

A d-semigroup/monoid is *twisted* if it satisfies the twisted law

[Manes 2006] refers to them as *guarded* semigroups/monoids

$d(x)$ is the *guard* of x

Twisted Domain Semigroups

Partial functions under composition satisfy another equational property called the *twisted law* by [Jackson and Stokes 2001]:

$$xd(y) = d(xy)x$$

This identity fails in $Rel(X)$ if we take x to be any relation that is not *deterministic*

A d-semigroup/monoid is *twisted* if it satisfies the twisted law

[Manes 2006] refers to them as *guarded* semigroups/monoids

$d(x)$ is the *guard* of x

Various *representation theorems* have been proved for families of semigroups with respect to partial functions. E.g.

• every *group* is embedded in the *symmetric group* $S(X)$ of all permutations of a set X.

• every *semigroup* is embedded in the *transformation semigroup* $T(X)$ of all functions on a set X.

Inverse semigroups are semigroups with a unary operation $^{-1}$ that satisfies the identities $x^{-1-1} = x$, $xx^{-1}x = x$ and $xx^{-1}yy^{-1} = yy^{-1}xx^{-1}$.

It is a standard result of semigroup theory (independently due to Vagner 1952 and Preston 1954) that

• every *inverse semigroup* is embedded in the *symmetric inverse semigroup* $PI(X)$ of all partial injections on X.

Various *representation theorems* have been proved for families of semigroups with respect to partial functions. E.g.

• every *group* is embedded in the *symmetric group* $S(X)$ of all permutations of a set X.

• every *semigroup* is embedded in the *transformation semigroup* $T(X)$ of all functions on a set X.

Inverse semigroups are semigroups with a unary operation $^{-1}$ that satisfies the identities $x^{-1-1} = x$, $xx^{-1}x = x$ and $xx^{-1}yy^{-1} = yy^{-1}xx^{-1}$.

It is a standard result of semigroup theory (independently due to Vagner 1952 and Preston 1954) that

• every *inverse semigroup* is embedded in the *symmetric inverse semigroup* $PI(X)$ of all partial injections on X.

Various *representation theorems* have been proved for families of semigroups with respect to partial functions. E.g.

• every *group* is embedded in the *symmetric group* $S(X)$ of all permutations of a set X.

• every *semigroup* is embedded in the *transformation semigroup* $T(X)$ of all functions on a set X.

Inverse semigroups are semigroups with a unary operation $^{-1}$ that satisfies the identities $x^{-1-1} = x$, $xx^{-1}x = x$ and $xx^{-1}yy^{-1} = yy^{-1}xx^{-1}$.

It is a standard result of semigroup theory (independently due to Vagner 1952 and Preston 1954) that

• every *inverse semigroup* is embedded in the *symmetric inverse semigroup* $PI(X)$ of all partial injections on X.

Various *representation theorems* have been proved for families of semigroups with respect to partial functions. E.g.

• every *group* is embedded in the *symmetric group* $S(X)$ of all permutations of a set X.

• every *semigroup* is embedded in the *transformation semigroup* $T(X)$ of all functions on a set X.

Inverse semigroups are semigroups with a unary operation $^{-1}$ that satisfies the identities $x^{-1-1} = x$, $xx^{-1}x = x$ and $xx^{-1}yy^{-1} = yy^{-1}xx^{-1}$.

It is a standard result of semigroup theory (independently due to Vagner 1952 and Preston 1954) that

• every *inverse semigroup* is embedded in the *symmetric inverse semigroup* $PI(X)$ of all partial injections on X.

Various *representation theorems* have been proved for families of semigroups with respect to partial functions. E.g.

• every *group* is embedded in the *symmetric group* $S(X)$ of all permutations of a set X.

• every *semigroup* is embedded in the *transformation semigroup* $T(X)$ of all functions on a set X.

Inverse semigroups are semigroups with a unary operation $^{-1}$ that satisfies the identities $x^{-1-1} = x$, $xx^{-1}x = x$ and $xx^{-1}yy^{-1} = yy^{-1}xx^{-1}$.

It is a standard result of semigroup theory (independently due to Vagner 1952 and Preston 1954) that

• every *inverse semigroup* is embedded in the *symmetric inverse semigroup* $PI(X)$ of all partial injections on X.

Various *representation theorems* have been proved for families of semigroups with respect to partial functions. E.g.

• every *group* is embedded in the *symmetric group* $S(X)$ of all permutations of a set X.

• every *semigroup* is embedded in the *transformation semigroup* $T(X)$ of all functions on a set X.

Inverse semigroups are semigroups with a unary operation $^{-1}$ that satisfies the identities $x^{-1-1} = x$, $xx^{-1}x = x$ and $xx^{-1}yy^{-1} = yy^{-1}xx^{-1}$.

It is a standard result of semigroup theory (independently due to Vagner 1952 and Preston 1954) that

• every *inverse semigroup* is embedded in the *symmetric inverse semigroup* $PI(X)$ of all partial injections on X.

Theorem (Trokhimenko 1973, Jackson Stokes 2001)

Every twisted d-semigroup can be embedded in a partial transformation semigroup, hence representable.

If the semigroup has a unit, it is mapped to the identity function.

Proof (outline).

Let S be a twisted d-semigroup and consider the partial transformation semigroup $PT(S)$. For $a \in S$ define

- $D_a = \{xd(a) : x \in S\} = \{y \in S : yd(a) = y\}$,
- $f_a : D_a \to S$ by $f_a(x) = xa$, and
- $h : S \to PT(S)$ by $h(a) = f_a$.

The map h is called the *Cayley embedding* and it remains to check that

1. $d(f_a) = f_{d(a)}$,
2. $f_a;f_b = f_{ab}$, and
3. h is injective.

This is fairly straight forward, using the twisted law for (2). □

Theorem (Trokhimenko 1973, Jackson Stokes 2001)

Every twisted d-semigroup can be embedded in a partial transformation semigroup, hence representable.
If the semigroup has a unit, it is mapped to the identity function.

Proof (outline).

Let S be a twisted d-semigroup and consider the partial transformation semigroup $PT(S)$. For $a \in S$ define

- $D_a = \{xd(a) : x \in S\} = \{y \in S : yd(a) = y\}$,
- $f_a : D_a \to S$ by $f_a(x) = xa$, and
- $h : S \to PT(S)$ by $h(a) = f_a$.

The map h is called the *Cayley embedding* and it remains to check that

1. $d(f_a) = f_{d(a)}$,
2. $f_a;f_b = f_{ab}$, and
3. h is injective.

This is fairly straight forward, using the twisted law for (2). □

Theorem (Trokhimenko 1973, Jackson Stokes 2001)

Every twisted d-semigroup can be embedded in a partial transformation semigroup, hence representable.
If the semigroup has a unit, it is mapped to the identity function.

Proof (outline).

Let S be a twisted d-semigroup and consider the partial transformation semigroup $PT(S)$. For $a \in S$ define

- $D_a = \{xd(a) : x \in S\} = \{y \in S : yd(a) = y\}$,
- $f_a : D_a \to S$ by $f_a(x) = xa$, and
- $h : S \to PT(S)$ by $h(a) = f_a$.

The map h is called the *Cayley embedding* and it remains to check that

1. $d(f_a) = f_{d(a)}$,
2. $f_a;f_b = f_{ab}$, and
3. h is injective.

This is fairly straight forward, using the twisted law for (2). □

So verification of deterministic sequential programs can be done abstractly entirely within the variety of twisted d-monoids

Corollary

Every commutative d-semigroup (d-monoid) is twisted, and can be embedded in a partial transformation semigroup (monoid)

$$d(xy)x \stackrel{D2}{=} d(xd(y))x \stackrel{com}{=} d(d(y)x)x \stackrel{D3}{=} d(y)d(x)x \stackrel{D1}{=} d(y)x \stackrel{com}{=} xd(y)$$

Hence the classes of commutative representable d-semigroups and d-monoids are both *finitely axiomatizable* varieties

This is in contrast to relation algebras where the variety of commutative representable relation algebras is *not finitely axiomatizable*

So verification of deterministic sequential programs can be done abstractly entirely within the variety of twisted d-monoids

Corollary

Every commutative d-semigroup (d-monoid) is twisted, and can be embedded in a partial transformation semigroup (monoid)

$$d(xy)x \stackrel{D2}{=} d(xd(y))x \stackrel{com}{=} d(d(y)x)x \stackrel{D3}{=} d(y)d(x)x \stackrel{D1}{=} d(y)x \stackrel{com}{=} xd(y)$$

Hence the classes of commutative representable d-semigroups and d-monoids are both *finitely axiomatizable* varieties

This is in contrast to relation algebras where the variety of commutative representable relation algebras is *not finitely axiomatizable*

So verification of deterministic sequential programs can be done abstractly entirely within the variety of twisted d-monoids

Corollary

Every commutative d-semigroup (d-monoid) is twisted, and can be embedded in a partial transformation semigroup (monoid)

$$d(xy)x \overset{D2}{=} d(xd(y))x \overset{com}{=} d(d(y)x)x \overset{D3}{=} d(y)d(x)x \overset{D1}{=} d(y)x \overset{com}{=} xd(y)$$

Hence the classes of commutative representable d-semigroups and d-monoids are both *finitely axiomatizable* varieties

This is in contrast to relation algebras where the variety of commutative representable relation algebras is *not finitely axiomatizable*

So verification of deterministic sequential programs can be done abstractly entirely within the variety of twisted d-monoids

Corollary

Every commutative d-semigroup (d-monoid) is twisted, and can be embedded in a partial transformation semigroup (monoid)

$$d(xy)x \overset{D2}{=} d(xd(y))x \overset{com}{=} d(d(y)x)x \overset{D3}{=} d(y)d(x)x \overset{D1}{=} d(y)x \overset{com}{=} xd(y)$$

Hence the classes of commutative representable d-semigroups and d-monoids are both *finitely axiomatizable* varieties

This is in contrast to relation algebras where the variety of commutative representable relation algebras is *not finitely axiomatizable*

So verification of deterministic sequential programs can be done abstractly entirely within the variety of twisted d-monoids

Corollary

Every commutative d-semigroup (d-monoid) is twisted, and can be embedded in a partial transformation semigroup (monoid)

$$d(xy)x \overset{D2}{=} d(xd(y))x \overset{com}{=} d(d(y)x)x \overset{D3}{=} d(y)d(x)x \overset{D1}{=} d(y)x \overset{com}{=} xd(y)$$

Hence the classes of commutative representable d-semigroups and d-monoids are both *finitely axiomatizable* varieties

This is in contrast to relation algebras where the variety of commutative representable relation algebras is *not finitely axiomatizable*

d-monoids are not finitely axiomatizable

Theorem (Hirsch and Mikulas 2009)

The class of representable domain (and antidomain) monoids is not finitely axiomatizable

Given a d-monoid or a-monoid A, they define a two-player game where the existential player has a *winning strategy* iff A is *representable*

Then they define an infinite sequence of d-monoids A_n that are *nonrepresentable* but the ultraproduct $(\prod A_n)/U$ for some nonprincipal ultrafilter U is *representable*

Free domain monoids

The free domain monoid is interesting for applications:

Identifies exactly those terms of domain monoids that have the same value in all domain monoids

A recursive description provides decision procedures

The domain axioms of domain monoids hold for relation algebras, domain quantales and stably supported quantales

The structure of free domain monoids can be lifted to free domain quantales

The free domain monoid will turn out to be representatble by binary relations

One-generated domain terms

(D1) $d(x)x = x$ (D2) $d(xd(y)) = d(xy)$ (D3) $d(d(x)y) = d(x)d(y)$

As usual, we define $x^0 = 1$ and $x^{n+1} = x^n x$

Lemma

In a domain monoid, if $m \leq n$ then

$$d(x^m)x^n = x^n \qquad \textit{and} \qquad d(x^m)d(x^n) = d(x^n)$$

Proof.

Assuming $m \leq n$, we write $x^n = x^m x^{n-m}$, and using (D1) we have

$$d(x^m)x^n = d(x^m)x^m x^{n-m} = x^m x^{n-m} = x^n$$

Now (D3) implies $d(x^m)d(x^n) = d(d(x^m)x^n) = d(x^n)$ □

Expanded normal forms

On elements of the form $d(x^j)$, the order is induced by the meet-semilattice structure: $d(x^j) \leq d(x^k)$ iff $j \geq k$, hence these elements form a chain

Can rewrite any term in *expanded normal form*:

$$d(x^{j_0})xd(x^{j_1})xd(x^{j_2})x\cdots xd(x^{j_m})$$

where $j_k \geq 1 + j_{k+1}$ for $k = 0, 1, \ldots, m-1$

E.g.
$d(x^4)x^2d(x) = d(x^4)xxd(x) = d(x^4)xd(x)xd(x) = d(x^4)xd(x^2)xd(x)$

where the last step holds since $d(xd(x)) = d(x^2)$ by (D2)

Decreasing sequences of numbers

For brevity denote such a term by the sequence $(j_0, j_1, j_2, \ldots, j_m)$

Note that this is always a strictly decreasing sequence of nonnegative integers

Let $\mathbf{P} = (P, \leq)$ be the set of all such sequences, ordered by reverse pointwise order

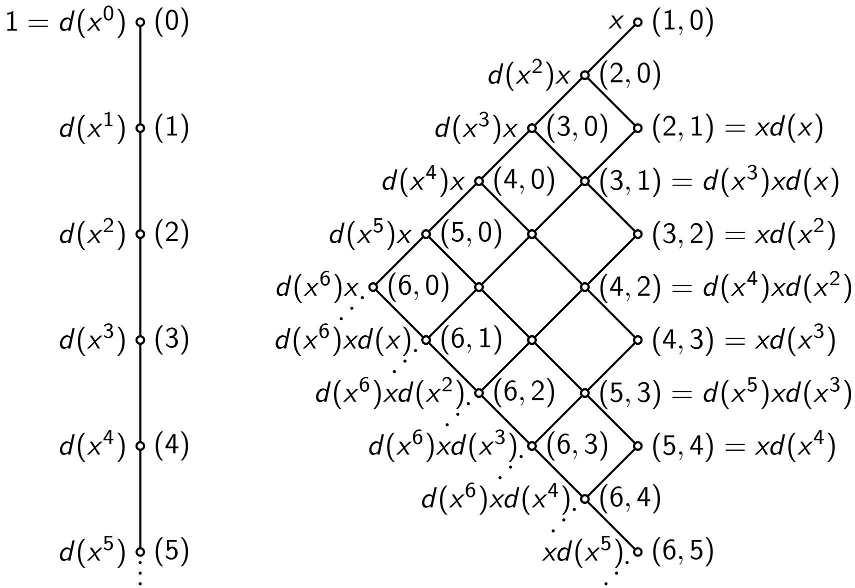
Thus sequences of different length are not comparable, and the maximal elements of this poset are

$$(0), \quad (1,0), \quad (2,1,0), \quad \ldots$$

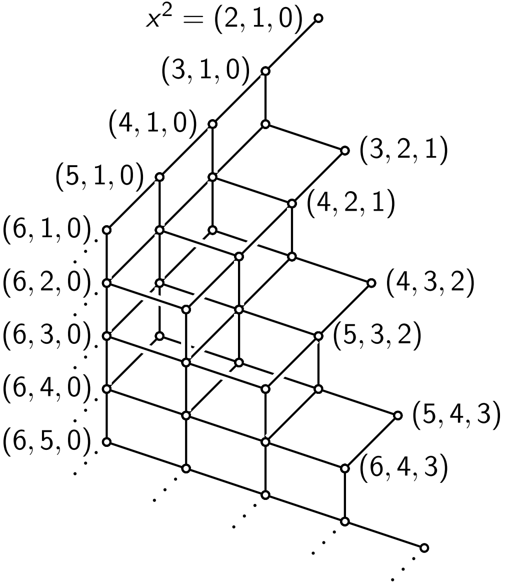
corresponding to the terms

$$d(1) = 1, \quad d(x)xd(1) = x, \quad d(x^2)xd(x)xd(1) = x^2, \quad \ldots$$

The poset of join-irreducibles below 1 and x



The poset of join-irreducibles below x^2



The product of two decreasing sequences

A multiplication is defined on P by the following "ripple product"

$$(j_0, j_1, j_2, \ldots, j_m) \cdot (k_0, k_1, k_2, \ldots, k_n) = (j'_0, j'_1, j'_2, \ldots, j'_m, k_1, k_2, \ldots, k_n)$$

where $j'_m = \max(j_m, k_0)$ and $j'_i = \max(j_i, j'_{i+1} + 1)$ for $i = m - 1, \ldots, 2, 1, 0$

For example, $(7, 3, 2) \cdot (4, 3, 1) = (7, 5, 4, 3, 1)$,

while $(4, 3, 1) \cdot (7, 3, 2) = (9, 8, 7, 3, 2)$

Can show that this is the result of multiplying the corresponding expanded normal forms and rewriting result in expanded normal form

It is tedious but not difficult to check that this operation is associative

Domain and partial order

The domain of a sequence $(j_0, j_1, j_2, \ldots, j_m)$ is the length-one sequence (j_0)

This corresponds to the domain term $d(x^{j_0})$

The following example indicates how elements of **P** can be represented by binary relations

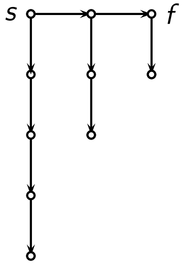
Example terms and corresponding relation

$$\mathbf{a} = (4, 2, 1)$$

$$t_{\mathbf{a}}(x) = d(x^4)xd(x^2)xd(x)$$

$$\mathbf{b} = (4, 3, 1)$$

$$t_{\mathbf{b}}(x) = d(x^4)xd(x^3)xd(x)$$



$$X_{\mathbf{a}} = \{\text{arrows}\}$$

Then $t_{\mathbf{a}}(X_{\mathbf{a}}) = \{(s, f)\}$ but $t_{\mathbf{b}}(X_{\mathbf{b}}) = \emptyset$

n-generated case (briefly)

A *d-term* is any term of the form $d(t)$

The identities $d(x)x = x$ (D1), $d(d(x)y) = d(x)d(y)$ (D3),

$d(d(x)) = x$, $d(e) = e$, $xe = x$ and $ex = x$

form a confluent and terminating rewrite system when applied from left to right (modulo associativity).

Will assume that all terms have been pre-normalized with respect to these identities, so non-constant terms contain no occurrance of e or of $\ldots d(d(\ldots)\ldots)\ldots$.

Hence every d-term is of the form

$$d(vd(t_1)\cdots d(t_{n-1})t_n) \quad \text{or} \quad d(vd(t_1)\cdots d(t_{n-1})d(t_n))$$

where v is a variable and the t_i are arbitrary pre-normalized terms.

In the first case, the term t_n is a variable or a product of terms that starts with a variable, so one application of $(D2)$ will convert it into the second form.

E.g. in the case $n = 1$, with $t_1 = xd(y)$ we have $d(vxd(y))$ converted to $d(vd(xd(y)))$.

This pre-normalized form for d-terms can be visualized by an edge-labelled tree, where the root has a single outgoing edge labelled v, and attached to the endpoint of this edge there are n (unordered) subtrees representing the d-terms $d(t_1), \ldots, d(t_n)$

Let v, w be variables and $s_1, \ldots, s_m, t_1, \ldots, t_n$ be terms.

Define a preorder relation $\sqsubseteq$ on pre-normalized d-terms as follows.

$$d(vd(s_1)\cdots d(s_m)) \sqsubseteq d(wd(t_1)\cdots d(t_n))$$

iff

$$v = w \quad \text{and} \quad \forall j \leq n\ \exists i \leq m\ (d(s_i) \sqsubseteq d(t_j))$$

This relation is obviously reflexive, and it is straight forward to check that it is transitive.

We say that two terms s, t are equal up to d-commutativity if t can be derived from s by a sequence of applications of $d(x)d(y) = d(y)d(x)$ (D4) (which is clearly reversible, hence this notion is an equivalence relation).

Lemma

If $d(s) \sqsubseteq d(t)$ and $d(t) \sqsubseteq d(s)$ then s and t are equal up to associativity and d-commutativity.

Lemma

If $d(s) \sqsubseteq d(t)$ then the identity $d(s)d(t) = d(s)$ holds in all domain monoids.

A d-term $d(vd(t_1)\cdots d(t_n))$ is *reduced* if $d(t_i) \not\sqsubseteq d(t_j)$ for all $i \neq j$

For example $d(xd(x)d(y))$ is reduced while $d(xd(xd(x)d(y))d(xd(y)))$ is not.

Theorem

Every d-term is equivalent to a reduced d-term, which is unique up to associativity and d-commutativity.

We now define the normal form for arbitrary domain monoid terms

A *d-sequence* is a finite product of d-terms. Note that 1 is considered a d-sequence given by the empty product. Hence any term t can be written in the form

$$t = s_0 v_1 s_1 v_2 s_2 \cdots s_{n-1} v_n s_n$$

where $v_1, \ldots, v_n$ are variables in X and $s_0, \ldots, s_n$ are d-sequences.

The term t is said to be in *expanded normal form* if for all $i < n$ some d-term $d(s)$ in the d-sequence s_i satisfies $d(s) \sqsubseteq d(v_{i+1} s_{i+1} \cdots v_n s_n)$.

Lemma

Suppose $t = s_0 v_1 s_1 v_2 s_2 \cdots s_{n-1} v_n s_n$ is in expanded normal form. Then $d(t) = s_0$.

We also extend the preorder $\sqsubseteq$ to d-sequences and arbitrary terms as follows. Let $d_1, \ldots, d_m, d'_1, \ldots, d'_n$ be d-terms. Then the $d_1 \cdots d_m \sqsubseteq d'_1 \cdots d'_n$ iff for all $j < n$ there exists $i < m$ such that $d_i \sqsubseteq d'_j$. In general, for alternations of d-sequences and variables we define

$$s_0 v_1 s_1 v_2 s_2 \cdots s_{m-1} v_m s_m \sqsubseteq t_0 w_1 t_1 w_2 t_2 \cdots t_{n-1} w_n t_n$$
$$\text{iff}$$
$$m = n, \quad v_i = w_i \text{ and } s_i \sqsubseteq t_i \text{ for all } i = 1, \ldots, n.$$

Theorem

Every term is equivalent to a term in expanded normal form, which is unique up to associativity and d-commutativity.

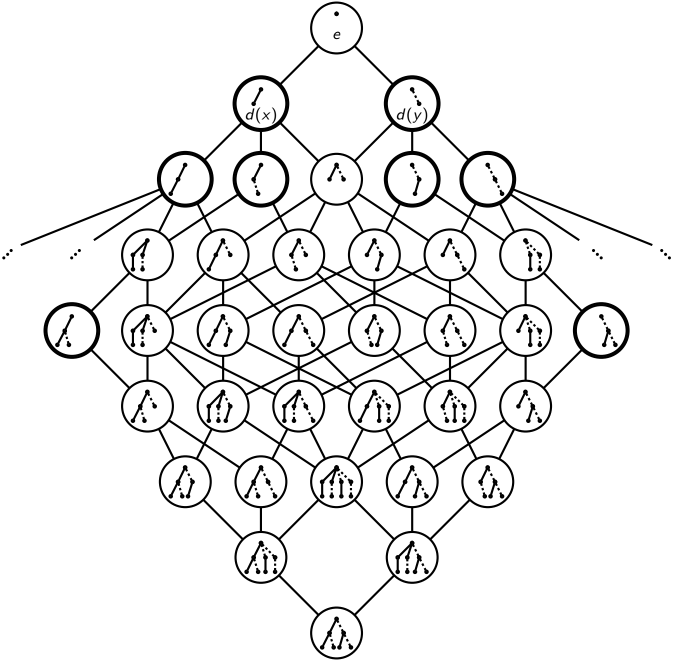


Figure: Below e in the meet semilattice $F_{\mathrm{DM}}(x, y)$

In the one-generated case, reduced d-terms are of the form $d(x^j)$, d-sequences can always be replaced by a single d-term, and the expanded normal form agrees with the one given before:

$$d(x^{j_0})xd(x^{j_1})xd(x^{j_2})x\cdots xd(x^{j_m}).$$

where $j_k > 1 + j_{k+1}$ for all $k = 0, \ldots, m-1$.

In the two-generated case, we note that $t = d(x^2)d(xy)$ and $s = d(xd(x)d(y))$ are both in expanded normal form, and that $s \leq t$.

To see that they represent distinct elements in the free algebra, we need to give an example of a domain monoid that distinguishes them.

Observe that the term $t(x, y) = d(x^2)d(xy)$ represents a tree with a root 0 and two branches with vertices $0, 1, 2$ and $0, 3, 4$.

The edges of the first branch are labeled by x and x, while the edges of the second branch are labeled by x and y.

Thus the tree defines a binary relation $X_t = \{(0,1),(1,2),(0,3)\}$ and another one $Y_t = \{(3,4)\}$.

It is easy to check that $t(X_t, Y_t) = \{(0,0)\}$, whereas $s(X_t, Y_t) = \emptyset$.

$\Rightarrow$ Relational representation similar to the one-generated case

Conclusion

The domain axioms (D1)-(D4) have appeared in computer science, relation algebra, semigroup theory and quantales

Every stably supported quantale and domain quantale contains a frame

Twisted d-monoids $=$ guarded monoids are representable

Representable d-monoids and a-monoids are not finitely axiomatizable (Hirsch-Mikulas 2009)

Free domain monoids can be described explicitly and are representable by binary relations

References

J. Desharnais, G. Struth, *Modal semirings revisited*. In P. Audebaud and C. Paulin-Mohring, editors, *MPC 2008*, volume 5133 of LNCS, pages 360–387. Springer, 2008

M. Hollenberg, *An equational axiomatization of dynamic negation and relational composition*, Journal of Logic, Language and Information, 6 (1997), 381–401

M. Jackson and T. Stokes, *An invitation to C-semigroups*, Semigroup Forum 62 (2001) 279–310

P. Jipsen and G. Struth, *The structure of the one-generated free domain semiring*, in "Relations and Kleene Algebra in Computer Science" (ed. R. Berghammer, B. Möller, G. Struth), LNCS, Vol. 4988, Springer (2008), 234–242

E. Manes, *Guarded and banded semigroups*, Semigroup Forum, Vol. 72, (2006), 94–120

P. Resende, *Étale groupoids and their quantales*, Advances in Mathematics, 208 (2007), 147–209

B. Möller, G. Struth, *Algebras of modal operators and partial correctness*, Theoretical Computer Science, 351, (2006), 221–239

B. M. Schein, *Relation algebras and function semigroups*, Semigroup Forum, Vol 1 (1970) 1–62

B. Schweizer and A. Sklar, *Function systems*, Math. Annalen, 172, (1967), 1–16

V. S. Trokhimenko, *Menger's function systems*, Izv. Vysš. Učebn. Zaved. Matematika, no. 11(138), (1973) 71–78