

Refinement
Algebra with
Explicit
Probabilism

T.M. Rabejaja
and J.W.
Sanders

From
Standard to
Probabilistic

Algebraic
Reasoning

The Algebra

A Folklore

Soundness

Proofs
Automation

Conclusion
and Future
Works

Refinement Algebra with Explicit Probabilism

T.M. Rabejaja and J.W. Sanders

July 29, 2009

Outline

- 1 From Standard to Probabilistic
- 2 Algebraic Reasoning
- 3 The Algebra
- 4 A Folklore
- 5 Soundness
- 6 Proofs Automation
- 7 Conclusion and Future Works

What we gain ?

- **Efficiency:** average running time with high probability [ex: Primality test].
- **Elegance:** syntactic simplicity though frequently non-intuitive [ex: Fractals].
- **Security:** modeling quantifiable unreliability, restricted demonic choice.
- ...

Tools

- PRAM or Probabilistic Turing Machine.
- Abstraction and Refinement: Denotational semantics.
- **Algebraic reasoning.**
- Model Checking, ...

Why?

- **Simplicity:** reasoning about loops are made algebraically.
- **Strength:** program transformations and refinement.
- **First order axioms** suggest automation.

How?

Refinement Algebra (RA)

$(X, \sqcap, \circ, *, \omega, \mathbb{I}, \top)$

[von Wright]



Probabilistic Demonic Refinement Algebra (pDRA)

$(X, \sqcap, \circ, *, \omega, \mathbb{I}, \top)$

[Meinecke and Solin]



Probabilistic Refinement Algebra (pRA)

$(X, \sqcap, \circ, \oplus, \cdot, *, \omega, \mathbb{I}, \top)$

[Rabehaja and Sanders]

So, ...

- Sound axiomatization.
- Direct control on probabilities.
- Probabilistic Kozen's "normal-form theorem".
- Equal efficiency of a loop and its normal form.
- Proofs automation through ATPs.

Standard axiomatization: $x, y, z \in X$

$$x \sqcap (y \sqcap z) = (x \sqcap y) \sqcap z$$

$$x \sqcap y = y \sqcap x$$

$$x \sqcap x = x$$

$$\top \sqcap x = x$$

$$x(y \sqcap z) = xy \sqcap xz$$

$$x^* = \mathbb{I} \sqcap xx^*$$

$$x \sqsubseteq yx \sqcap z \implies x \sqsubseteq y^*z$$

$$x \sqsubseteq x(y \sqcap \mathbb{I}) \sqcap z \implies x \sqsubseteq zy^*$$

$$x(yz) = (xy)z$$

$$\mathbb{I}x = x = x\mathbb{I}$$

$$\top x = \top$$

$$(x \sqcap y)z = xz \sqcap yz$$

$$x^\omega = \mathbb{I} \sqcap xx^\omega$$

$$yx \sqcap z \sqsubseteq x \implies y^\omega z \sqsubseteq x$$

$$xy \sqcap z \sqsubseteq x \implies zy^\omega \sqsubseteq x$$

Probabilistic Demonic axiomatization: $x, y, z \in X$

$$x \sqcap (y \sqcap z) = (x \sqcap y) \sqcap z$$

$$x \sqcap y = y \sqcap x$$

$$x \sqcap x = x$$

$$\top \sqcap x = x$$

$$x(y \sqcap z) \sqsubseteq xy \sqcap xz$$

$$x^* = \mathbb{I} \sqcap xx^*$$

$$x \sqsubseteq yx \sqcap z \implies x \sqsubseteq y^*z$$

$$x \sqsubseteq x(y \sqcap \mathbb{I}) \sqcap z \implies x \sqsubseteq zy^*$$

$$x(yz) = (xy)z$$

$$\mathbb{I}x \sqsubseteq x = x\mathbb{I}$$

$$\top x = \top$$

$$(x \sqcap y)z = xz \sqcap yz$$

$$x^\omega = \mathbb{I} \sqcap xx^\omega$$

$$yx \sqcap z \sqsubseteq x \implies y^\omega z \sqsubseteq x$$

The fact that an adversary cannot access to the result of a coin flip before the process itself is captured by the *subdistributivity*.

Probabilistic axiomatization: $x, y, z \in X$ and $p, q \in \Lambda$

$$x \sqcap (y \sqcap z) = (x \sqcap y) \sqcap z$$

$$x \sqcap y = y \sqcap x$$

$$x \sqcap x = x$$

$$\top \sqcap x = x$$

$$x(y \sqcap z) \sqsubseteq xy \sqcap xz$$

$$x^* = \mathbb{I} \sqcap xx^*$$

$$x \sqsubseteq yx \sqcap z \implies x \sqsubseteq y^*z$$

$$x \sqsubseteq x(y \sqcap \mathbb{I}) \sqcap z \implies x \sqsubseteq zy^*$$

$$x(yz) = (xy)z$$

$$\mathbb{I}x = x = x\mathbb{I}$$

$$\top x = \top$$

$$(x \sqcap y)z = xz \sqcap yz$$

$$x^\omega = \mathbb{I} \sqcap xx^\omega$$

$$yx \sqcap z \sqsubseteq x \implies y^\omega z \sqsubseteq x$$

Plus ...

$$\begin{aligned}x \oplus (y \oplus z) &= (x \oplus y) \oplus z \\x \oplus (y \sqcap z) &= (x \oplus y) \sqcap (x \oplus z) \\p(qx) &= (pq)x \\1x &= x \\(p + q)x &= px \oplus qx\end{aligned}$$

$$\begin{aligned}x \oplus y &= y \oplus x \\(y \oplus z)x &= yx \oplus zx \\p(xy) &= (px)y \\0x &= \perp \\p(qx \oplus ry) &= pqx \oplus pry\end{aligned}$$

As usual, $\perp = \mathbb{I}^\omega$.

Some consequences of the axioms ...

Theorem

If $x, y, z \in X$ and $p \in \Lambda$ then

- 1 $x \sqcap y \sqsubseteq px \oplus \bar{p}y$,
- 2 $x(yx)^\omega \sqsubseteq (xy)^\omega x$.
- 3 $(x \sqcap y)^\omega = x^\omega (yx^\omega)^\omega$,

- From 1, (\oplus, \cdot) defines a stronger form of non-determinism.
- If x is conjunctive¹ then 2 becomes an equality.
- Leap-frog inherited from Refinement Algebra.

¹ $x(y \sqcap z) = xy \sqcap xz$ for any $y, z \in X$

Normal-Form Programs

Given a structured (probabilistic) program

$$x_1 \circ (\mathbf{if} \ b_1 \rightarrow x_2 \circ \ \mathbf{do} \ b_2 \rightarrow x_3 \circ \cdots \ \mathbf{od} \\ \overline{b_1} \rightarrow (y_2 \ [\frac{1}{2}] \ y_3) \circ \cdots \ \mathbf{fi}) \circ z$$

is there a “normal-form” (probabilistic) program

$$x \circ \ \mathbf{do} \ b \rightarrow y \ \mathbf{od}, \ x \text{ and } y \text{ are } \textit{loop-free}$$

which simulates (and is simulated) by the above program?

Algebraically, find \star -free terms x, y such that

$$x_1(b_1[x_2(x_3 \cdots)]^\star) \sqcap \overline{b_1}[(\frac{1}{2}y_2 \oplus \frac{1}{2}y_3) \cdots]z =_S xy^\star$$

where S is a finite set of guards checking some fresh Boolean variables and $x^\star = (bx)^\omega \overline{b}$ for some guard b .

Probability-free setting

- Kleene algebras with tests gives an algebraic framework to prove the results within *partial correctness* [Kozen].
- Refinement algebra works within *total correctness* [Solin].
- Equal efficiency of a program and its normal-form version.

To overcome

- Kozen and Solin's proofs depend upon the distributivity $x(y \sqcap z) = xy \sqcap xz$.
- Introduce probabilistic branching.

Fortunately

- The four transformations of Kozen holds for Probabilistic Demonic Refinement Algebra:

Lemma

Let $u, v, x, y \in X$ and b, c be guards, then

$$\textcircled{1} \quad bux^* \sqcap \bar{b}vy^* =_c (cu \sqcap \bar{c}v)(cx \sqcap \bar{c}y)^*$$

$$\textcircled{2} \quad (xy^*)^* = bx(cy \sqcap \bar{c}x)^* \sqcap \bar{b}$$

$$\textcircled{3} \quad x^*y =_c c(x(c \sqcap \bar{c}y))^* \sqcap \bar{c}y$$

- $p(sbx) \oplus \bar{p}(s'\bar{b}y) = (ps \oplus \bar{p}s')(bx \sqcap \bar{b}y)$ where $s = [b := \text{true}]$ and $s' = [b := \text{false}]$.

1-bounded Expectation Transformers

Given a set of states S , $\mathbb{E}_1 : S \rightarrow [0, 1]$, the model is given by

$$\mathbb{T}_1 : \mathbb{E}_1 \rightarrow \mathbb{E}_1$$

where $\Lambda : S \rightarrow [0, 1]$ and

$$[\top] = \lambda e : \mathbb{E}_1 \cdot \mathbf{1}$$

$$[\mathbb{I}] = \lambda e : \mathbb{E}_1 \cdot e$$

$$[P \circledast Q] = [P] \circ [Q]$$

$$[P \sqcap Q] = [P] \min [Q]$$

$$[P \oplus Q] = [P] + [Q]$$

$$[p \cdot P] \cdot e = \lambda s : S \cdot ([p] \cdot s)([P] \cdot e \cdot s)$$

* and ω are interpreted as respectively *greatest* and *least fixpoints*.

Other models

Given a model \mathcal{M} of pDRA with provided with the operators x_p^\oplus , one may generate a model of pRA.

Construction:

1. $\pi : \mathcal{M} \times \Lambda \rightarrow \mathcal{M}$, $(x, p) \mapsto x_p^\oplus \perp$, where $\Lambda = S \rightarrow [0, 1]$
2. $\varepsilon : \mathcal{M} \rightarrow \mathcal{M} \times \Lambda$, $x \mapsto (x, \mathbf{1})$, (π, ε) is a *Galois embedding*

One transfers constant and unary function with (π, ε) .

3. Provide $\mathcal{M} \times \Lambda$ with operators:

$$(x, p) \circledast (y, q) := (x \circledast \pi(y, q), p)$$

$$(x, p) \sqcap (y, q) := \left(\pi \left(x, \frac{p}{p \sqcup q} \right) \sqcap \pi \left(y, \frac{q}{p \sqcup q} \right), p \sqcup q \right)$$

$$(x, p) \oplus (y, q) := \left(x \frac{p}{p+q} \oplus y, p+q \right)$$

$$p \cdot (x, q) := (x, pq)$$

4. Construct a congruence on $\mathcal{M} \times \Lambda$:

$$(x, p) \sim (y, q)$$

iff

$$\forall r : \Lambda \cdot p \sqcup q \leq r \wedge r > \mathbf{0} \implies \pi(x, p/r) = \pi(y, q/r)$$

Theorem (soundness)

$(\mathcal{M} \times \Lambda / \sim, \sqcap, \circ, \oplus, \cdot, \top, \mathbb{I})$ is a probabilistic refinement algebra

- Most of the properties of the operators are proved in $\mathcal{M} \times \Lambda$ which generate similar proofs within the quotient $\mathcal{M} \times \Lambda / \sim$ since \sim is a congruence.
- Necessity of \sim : some properties such as $\mathbb{I} \circ x = x$ does not holds in the simple product.

- First order axioms enable an automatic proof of refinements and program transformation.

EXAMPLE. Most of our results has been proved automatically using Prover9.

InputFile: axioms.in

```
% Probabilistic Demonic Refinement Algebra
formulas(sos).
x + y = y + x.
x + (y + z) = (x + y) + z.
x + x = x.
x + T = x.
x ; (y ; z) = (x ; y) ; z.
(x ; I = x) & (I ; x = x).
T ; x = T.
(x <= y) <-> (x + y = x).
x ; (y + z) <= x ; y + x ; z.
(x + y) ; z = x ; z + y ; z.
x* = I + x ; x*.
((y ; x + z) <= x) -> (y* ; z <= x).
end_of_list.
```

```
% Bottom element
formulas(sos).
  1* = B.
end_of_list.

% Probability
formulas(sos).
  x @ (y @ z) = (x @ y) @ z.
  x @ y = y @ x.
  x @ (y + z) = (x @ y) + (x @ z).
  (x @ y) ; z = x ; z @ y ; z.
  p^(q^x) = (p^q)^x.
  p^(x ; y) = (p^x) ; y.
  1^x = x.
  0^x = B.
  (p @ q)^x = (p^x) @ (q^x).
% p^(q^x @ r^y) = (p^q)^x @ (p^r)^y.
end_of_list.

% demonic choice <= probabilistic choice.
formulas(goals).
  p @ q = 1 -> x + y <= p^x @ q^(x + y).
% p @ q = 1 -> x + y <= p^x @ q^y.
end_of_list.
```

Summary

- Probabilistic programming is powerful.
- pRA: Algebraic tool for reasoning about probabilistic computations.
- Soundness: models constructed upon the usual semantics.
- Probabilistic normal form theorem.
- Automation of proofs in pRA

Further works

- Algebraic proof of the equal efficiency of a probabilistic program and its normal-form version.
- Nature of the two form of probabilistic choice.
- Further attention into automation.