Models of Randomness

Part I: a sketchy survey

Fritz Obermeyer

Department of Mathematics
Carnegie-Mellon University

2008:04:08

Outline

Motivational outline

Want to prove equivalence between randomized algorithms.

Motivational outline

Want to prove equivalence between randomized algorithms.

Need a programming language with random monad;

Motivational outline

Want to prove equivalence between randomized algorithms.

Need a programming language with random monad; and a domain-theoretic model of this language.

Motivational outline

Want to prove equivalence between randomized algorithms.

Need a programming language with random monad; and a domain-theoretic model of this language.

This talk surveys some attempts at models.

Motivational outline

Want to prove equivalence between randomized algorithms.

Need a programming language with random monad; and a domain-theoretic model of this language.

This talk surveys some attempts at models.

Later Part II tries to find a random monad
in a type-as-ambiguity framework (closures).

Bayesian networks

Consider entirely first-order programs, with no looping/recursion,

Bayesian networks

Consider entirely first-order programs, with no looping/recursion, e.g.,

sample x from unif$(0, 1)$ in
sample y from unif$(0, \mathsf{x})$ in
sample z from unif$(\mathsf{x}, 0)$ in
if $\mathsf{y} + \mathsf{z} < 1/2$ then 0 else 1

Bayesian networks

Consider entirely first-order programs, with no looping/recursion, e.g.,

sample x from unif$(0, 1)$ in
sample y from unif$(0, \mathsf{x})$ in
sample z from unif$(\mathsf{x}, 0)$ in
if $\mathsf{y} + \mathsf{z} < 1/2$ then 0 else 1

(notice we can forget x after sampling y,z)

Bayesian networks

Consider entirely first-order programs, with no looping/recursion, e.g.,

sample x from unif$(0, 1)$ in
sample y from unif$(0, x)$ in
sample z from unif$(x, 0)$ in
if $y + z < 1/2$ then 0 else 1

(notice we can forget x after sampling y,z)
This is a Bayesian network (see picture).

Bayesian networks

Consider entirely first-order programs, with no looping/recursion, e.g.,

sample x from unif$(0, 1)$ in
sample y from unif$(0, x)$ in
sample z from unif$(x, 0)$ in
if $y + z < 1/2$ then 0 else 1

(notice we can forget x after sampling y,z)
This is a Bayesian network (see picture).
All Bayesian networks are so expressible.

Markov chains

Consider iterative loops,

Markov chains

Consider iterative loops, e.g.

x $\leftarrow$ sample x0 from normal$(0,1) \in$ x0.
while $\ldots$:
    sample n from normal$(0,1)$ in
    let x$' = 1$ $+$ x$/2$ in
    x $\leftarrow$ x$'$ $+$ n

Markov chains

Consider iterative loops, e.g.

$\mathsf{x} \leftarrow$ sample x0 from normal$(0,1) \in$ x0.
while ... :
    sample n from normal$(0,1)$ in
    let $\mathsf{x}' = 1 + \mathsf{x}/2$ in
    $\mathsf{x} \leftarrow \mathsf{x}' + \mathsf{n}$

This is a Markov chain (see picture).

Markov chains

Consider iterative loops, e.g.

x $\leftarrow$ sample x0 from normal$(0, 1) \in$ x0.
while $\ldots$:
    sample n from normal$(0, 1)$ in
    let x$' = 1$ $+$ x$/2$ in
    x $\leftarrow$ x$'$ $+$ n

This is a Markov chain (see picture).
All Markov chains are so expressible.

Prototype for a probability theory

- set of states/points (maybe with structure)

Prototype for a probability theory

- set of states/points (maybe with structure)
- space of events/predicates

Prototype for a probability theory

- set of states/points (maybe with structure)
- space of events/predicates
- morphisms between state-event spaces

Prototype for a probability theory

- set of states/points (maybe with structure)
- space of events/predicates
- morphisms between state-event spaces
- random monad = probability distributions

Prototype for a probability theory

- set of states/points (maybe with structure)
- space of events/predicates
- morphisms between state-event spaces
- random monad = probability distributions
- extra structure: products, exponentials, ...

We start with finite sets,

Prototype for a probability theory

- set of states/points (maybe with structure)
- space of events/predicates
- morphisms between state-event spaces
- random monad = probability distributions
- extra structure: products, exponentials, ...

We start with finite sets,
generalize to probability measures,

Prototype for a probability theory

- set of states/points (maybe with structure)
- space of events/predicates
- morphisms between state-event spaces
- random monad = probability distributions
- extra structure: products, exponentials, ...

We start with finite sets,
generalize to probability measures, then
weaken the event language,

Prototype for a probability theory

- set of states/points (maybe with structure)
- space of events/predicates
- morphisms between state-event spaces
- random monad = probability distributions
- extra structure: products, exponentials, ...

We start with finite sets,
generalize to probability measures, then
weaken the event language,
add structure among points,

Prototype for a probability theory

- set of states/points (maybe with structure)
- space of events/predicates
- morphisms between state-event spaces
- random monad = probability distributions
- extra structure: products, exponentials, ...

We start with finite sets,
generalize to probability measures, then
weaken the event language,
add structure among points, and
end with fully algebraic approaches.

Finite sets: states, morphisms, extra structure

Start with a finite set X of states.

Finite sets: states, morphisms, extra structure

Start with a finite set X of states.
No need for event spacee.

Finite sets: states, morphisms, extra structure

Start with a finite set X of states.
No need for event spacee.

Morphisms are just functions,

Finite sets: states, morphisms, extra structure

Start with a finite set X of states.
No need for event spacee.

Morphisms are just functions,
product, exponenial are standard.

Finite sets: states, morphisms, extra structure

Start with a finite set X of states.
No need for event spacee.

Morphisms are just functions,
product, exponenial are standard.
NO recursive types,

Finite sets: states, morphisms, extra structure

Start with a finite set X of states.
No need for event spacee.

Morphisms are just functions,
product, exponenial are standard.
NO recursive types, NO infinite types

Finite sets: Random functor?

Like the powerset, Rand is functorial,

Finite sets: Random functor?

Like the powerset, Rand is functorial,
On objects: $\mathsf{Rand}((\mathsf{X}, \mathbf{F})) = (\mathsf{X}', \mathbf{F}')$ where

- $\mathsf{X}' \;= $ probability measures over $(\mathsf{X}, \mathbf{F})$

Finite sets: Random functor?

Like the powerset, Rand is functorial,
On objects: $\mathsf{Rand}((X, \mathbf{F})) = (X', \mathbf{F}')$ where

- $X' = $ probability measures over $(X, \mathbf{F})$
- $\mathbf{F}' = $ generated by (right?)

$$\{\ \{\ f \in X' \ \mid\ f^{-1}B \ \subseteq\ A\ \}\ \mid\ A \in \mathbf{F},\ B \in G\ \}$$

Finite sets: Random functor?

Like the powerset, Rand is functorial,
On objects: $\text{Rand}((X, \mathbf{F})) = (X', \mathbf{F}')$ where

- $X' = $ probability measures over $(X, \mathbf{F})$
- $\mathbf{F}' = $ generated by (right?)

$$\{ \{ f \in X' \mid f^{-1}B \subseteq A \} \mid A \in \mathbf{F}, B \in G \}$$

On arrows: for $h : (X, \mathbf{F}) \rightarrow (\mathbf{Y}, G)$,
$\text{Rand}(h) = h' : (X', \mathbf{F}') \rightarrow (\mathbf{Y}', G')$ where for $p : X'$ a probability measure on $(X, \mathbf{F})$, $B \in G$,

$$\text{Rand}(h)(p)(B) = p(h^{-1}B)$$

Finite sets: Random functor?

Like the powerset, Rand is functorial,
On objects: $\mathrm{Rand}((X, \mathbf{F})) = (X', \mathbf{F}')$ where

- $X' \;=$ probability measures over $(X, \mathbf{F})$
- $\mathbf{F}' \;=$ generated by (right?)

$$\{\ \{\ f \in X' \ \mid\ f^{-1}B \ \subseteq\ A\ \}\ \mid\ A \in \mathbf{F},\ B \in G\ \}$$

On arrows: for $h : (X, \mathbf{F}) \to (\mathbf{Y}, G)$,
$\mathrm{Rand}(h) = h' : (X', \mathbf{F}') \to (\mathbf{Y}', G')$ where for $p : X'$ a probability measure on $(X, \mathbf{F})$, $B \in G$,

$$\mathrm{Rand}(h)(p)(B) = p(h^{-1}B)$$

But random functor doesn't land in finite sets.

Random monad (pieces)

The probability functor forms a monad with natural

$$\begin{aligned} \mathsf{always} &: \forall \mathsf{a}.\ \mathsf{a} \rightarrow \mathsf{Rand}\ \mathsf{a} \\ \mathsf{mix} &: \forall \mathsf{a}.\ \mathsf{Rand}(\mathsf{Rand}\ \mathsf{a}) \rightarrow \mathsf{Rand}\ \mathsf{a} \end{aligned}$$

Random monad (pieces)

The probability functor forms a monad with natural

$$\text{always} : \forall a.\ a \rightarrow \text{Rand}\ a$$
$$\text{mix} : \forall a.\ \text{Rand}(\text{Rand}\ a) \rightarrow \text{Rand}\ a$$

equivalently a Kleisli triple with 'always' and

$$\text{sample} : \forall a, b.\ \text{Rand}\ a \rightarrow (a \rightarrow \text{Rand}\ b) \rightarrow \text{Rand}\ b$$

Random monad (pieces)

The probability functor forms a monad with natural

$$\begin{aligned} \text{always} &: \forall a.\ a \to \text{Rand}\ a \\ \text{mix} &: \forall a.\ \text{Rand}(\text{Rand}\ a) \to \text{Rand}\ a \end{aligned}$$

equivalently a Kleisli triple with 'always' and

$$\text{sample} : \forall a, b.\ \text{Rand}\ a \to (a \to \text{Rand}\ b) \to \text{Rand}\ b$$

In finite-sets, semantics is

$$[\text{always}\ x](y) = \delta_{x,y}$$
$$[\text{mix}\ p](x) = \int [p](q)\ q(x)\ dq$$

oops: p infinite

$$[\text{sample}\ p\ f](y) = \sum_x [p](x)\ [f](y)(x)$$

Random monad (properties)

Being a monad requires equations

```
sample x from p in always x
```

Random monad (properties)

Being a monad requires equations

sample x from p in always x = p,

Random monad (properties)

Being a monad requires equations

$$\text{sample } x \text{ from } p \text{ in always } x = p,$$
$$\text{sample } x \text{ from always } y \text{ in } f\ x$$

Random monad (properties)

Being a monad requires equations

$$\text{sample } x \text{ from } p \text{ in always } x = p,$$
$$\text{sample } x \text{ from always } y \text{ in } f\ x = f\ y,$$

Random monad (properties)

Being a monad requires equations

$$\begin{aligned}
&\text{sample } x \text{ from } p \text{ in always } x = p,\\
&\text{sample } x \text{ from always } y \text{ in } f\ x = f\ y,\\
&\text{sample } y \text{ from } (\text{sample } x \text{ from } p \text{ in } f\ x) \text{ in } g\ y
\end{aligned}$$

Random monad (properties)

Being a monad requires equations

$$\text{sample } x \text{ from } p \text{ in always } x = p,$$
$$\text{sample } x \text{ from always } y \text{ in } f\ x = f\ y,$$
$$\begin{aligned}&\text{sample } y \text{ from (sample } x \text{ from } p \text{ in } f\ x) \text{ in } g\ y\\ &\quad = \text{sample } x \text{ from } p \text{ in}\\ &\qquad \text{sample } y \text{ from } f\ x \text{ in}\\ &\qquad g\ y\end{aligned}$$

Random monad (properties)

Being a monad requires equations

```
sample x from p in always x = p,
sample x from always y in f x = f y,
sample y from (sample x from p in f x) in g y
      =  sample x from p in
         sample y from f x in
         g y
```

Being a computational monad (a la Moggi) requires also:

- 'always' is mono
- monad plays nicely with products and sums

Probability measures: states, events, morphisms

Start with a state set X (unstructured).

Probability measures: states, events, morphisms

Start with a state set X (unstructured).
The event space is a sigma-algebra F of X,

Probability measures: states, events, morphisms

Start with a state set X (unstructured).
The event space is a sigma-algebra F of X,
a structure $\langle \mathsf{X},\ \mathbf{F} \subseteq \mathcal{P}(\Omega),\ \neg,\ \bigvee^{\leq\omega} \rangle$,

Probability measures: states, events, morphisms

Start with a state set X (unstructured).
The event space is a sigma-algebra F of X,
a structure $\langle \mathsf{X},\ \mathbf{F} \subseteq \mathcal{P}(\Omega),\ \neg,\ \bigvee^{\leq\omega} \rangle$,
where $\bot = \bigvee\ \emptyset,\ \top = \bigvee\ \mathbf{F}$ are definable.

Probability measures: states, events, morphisms

Start with a state set X (unstructured).
The event space is a sigma-algebra F of X,
a structure $\langle \mathsf{X}, \ \mathbf{F} \subseteq \mathcal{P}(\Omega), \ \neg, \ \bigvee^{\leq\omega} \rangle$,
where $\bot = \bigvee \ \emptyset, \ \top = \bigvee \ \mathbf{F}$ are definable.

A morphism is a sigma-algebra hom,

Probability measures: states, events, morphisms

Start with a state set X (unstructured).
The event space is a sigma-algebra F of X,
a structure $\langle \mathsf{X},\ \mathbf{F} \subseteq \mathcal{P}(\Omega),\ \neg,\ \bigvee^{\leq\omega} \rangle$,
where $\bot = \bigvee\ \emptyset,\ \top = \bigvee\ \mathbf{F}$ are definable.

A morphism is a sigma-algebra hom, a measurable function $\mathsf{f}\colon\ \mathsf{X} \to \mathbf{Y}$, whose preimage induces a hom

$$\mathsf{f}'\ :\ \langle \mathbf{F}, \neg, \biguplus \rangle\ \leftarrow\ \langle \mathsf{G}, \neg, \biguplus \rangle$$

Probability measures: states, events, morphisms

Start with a state set X (unstructured).
The event space is a sigma-algebra F of X,
a structure $\langle \mathsf{X},\ \mathbf{F} \subseteq \mathcal{P}(\Omega),\ \neg,\ \bigvee^{\leq\omega}\rangle$,
where $\bot = \bigvee\ \emptyset,\ \top = \bigvee\ \mathbf{F}$ are definable.

A morphism is a sigma-algebra hom, a measurable function $\mathsf{f}\colon\ \mathsf{X} \to \mathbf{Y}$, whose preimage induces a hom

$$\mathsf{f}'\ :\ \langle \mathbf{F}, \neg, \biguplus \rangle\ \leftarrow\ \langle \mathsf{G}, \neg, \biguplus \rangle$$

Measures: Random states

A random state is a probability measure, a hom

$$\langle \mathbf{F}, \emptyset, \mathsf{X}, \biguplus^{\omega} \rangle \longrightarrow \langle [0,1], 0, 1, \sum^{\omega} \rangle$$

Measures: Random states

A random state is a probability measure, a hom

$$\langle \mathbf{F}, \emptyset, \mathsf{X}, \biguplus^{\omega} \rangle \quad \longrightarrow \quad \langle [0,1], 0, 1, \sum^{\omega} \rangle$$

i.e., functions p satisfying

$$\mathsf{p}(\bot) = 0$$
$$\mathsf{p}(\top) = 1$$
$$\mathsf{p}(\biguplus_{\mathsf{i}} \ \mathsf{A_i}) = \sum_{\mathsf{i}} \ \mathsf{p}(\mathsf{A_i})$$

Measures: Random states

A random state is a probability measure, a hom

$$\langle \mathbf{F}, \emptyset, \mathsf{X}, \biguplus^{\omega} \rangle \longrightarrow \langle [0,1], 0, 1, \sum^{\omega} \rangle$$

i.e., functions p satisfying

$$p(\bot) = 0$$
$$p(\top) = 1$$
$$p(\biguplus_i A_i) = \sum_i p(A_i)$$

and hence $p(\neg A) = 1 - p(A)$

Question equivalent to additivity+continuity?

Measures: extra structure

Product sigma-algebras are generaged by rectangles

Measures: extra structure

Product sigma-algebras are generaged by rectangles
Exponentials have pointwise sigma-algebra (right?)

Measures: extra structure

Product sigma-algebras are generaged by rectangles
Exponentials have pointwise sigma-algebra (right?)

NO untyped/unityped model of lambda-calculus

Probability valuations

Relax event logic from sigma-algebra to topology;

Probability valuations

Relax event logic from sigma-algebra to topology;
abstract away points to frames/locales/CHAs.

Probability valuations

Relax event logic from sigma-algebra to topology;
abstract away points to frames/locales/CHAs.

Definition

A probability valuation on F is a monotone $\mathsf{p} : \mathbf{F} \to [0, 1]$ satisfying

$$\mathsf{p}(\bot) = 0,$$

Probability valuations

Relax event logic from sigma-algebra to topology;
abstract away points to frames/locales/CHAs.

Definition

A probability valuation on F is a monotone $\mathsf{p} : \mathbf{F} \to [0, 1]$ satisfying

$$\mathsf{p}(\bot) = 0, \quad \mathsf{p}(\top) = 1$$

Probability valuations

Relax event logic from sigma-algebra to topology;
abstract away points to frames/locales/CHAs.

Definition

A probability valuation on F is a monotone $\mathsf{p} : \mathbf{F} \to [0, 1]$ satisfying

$$\mathsf{p}(\bot) = 0, \quad \mathsf{p}(\top) = 1$$
$$\mathsf{p}(\mathsf{A}) + \mathsf{p}(\mathsf{B}) = \mathsf{p}(\mathsf{A} \sqcap \mathsf{B}) + \mathsf{p}(\mathsf{A} \sqcup \mathsf{B})$$

Probability valuations

Relax event logic from sigma-algebra to topology;
abstract away points to frames/locales/CHAs.

Definition

A probability valuation on F is a monotone $p : \mathbf{F} \to [0,1]$ satisfying

$$p(\bot) = 0, \quad p(\top) = 1$$
$$p(A) + p(B) = p(A \sqcap B) + p(A \sqcup B)$$

we also assume continuity (some authors don't).
(analogous to countable additivity?)

Probability valuations

Relax event logic from sigma-algebra to topology;
abstract away points to frames/locales/CHAs.

Definition

A probability valuation on F is a monotone $\mathsf{p} : \mathbf{F} \to [0,1]$ satisfying

$$\mathsf{p}(\bot) = 0, \quad \mathsf{p}(\top) = 1$$
$$\mathsf{p}(\mathsf{A}) + \mathsf{p}(\mathsf{B}) = \mathsf{p}(\mathsf{A} \sqcap \mathsf{B}) + \mathsf{p}(\mathsf{A} \sqcup \mathsf{B})$$

we also assume continuity (some authors don't).
(analogous to countable additivity?)

...extension theorems

Probability valuations

Relax event logic from sigma-algebra to topology;
abstract away points to frames/locales/CHAs.

Definition

A probability valuation on F is a monotone $\mathsf{p} : \mathbf{F} \to [0, 1]$ satisfying

$$\mathsf{p}(\bot) = 0, \quad \mathsf{p}(\top) = 1$$
$$\mathsf{p}(\mathsf{A}) + \mathsf{p}(\mathsf{B}) = \mathsf{p}(\mathsf{A} \sqcap \mathsf{B}) + \mathsf{p}(\mathsf{A} \sqcup \mathsf{B})$$

we also assume continuity (some authors don't).
(analogous to countable additivity?)

...extension theorems e.g.

Probability valuations

Relax event logic from sigma-algebra to topology;
abstract away points to frames/locales/CHAs.

Definition

A probability valuation on F is a monotone $\mathsf{p}:\mathbf{F}\to[0,1]$ satisfying

$$\mathsf{p}(\bot) = 0, \quad \mathsf{p}(\top) = 1$$
$$\mathsf{p}(\mathsf{A}) + \mathsf{p}(\mathsf{B}) = \mathsf{p}(\mathsf{A} \sqcap \mathsf{B}) + \mathsf{p}(\mathsf{A} \sqcup \mathsf{B})$$

we also assume continuity (some authors don't).
(analogous to countable additivity?)

...extension theorems e.g.

Theorem

(Jones) Every continuous valuation on a continuous dcpo

Directed-complete partial orders.

Start with a dcpo of states.

Directed-complete partial orders.

Start with a dcpo of states.
Events are Scott-open sets (a frame).

Directed-complete partial orders.

Start with a dcpo of states.
Events are Scott-open sets (a frame).
Morphisms are Scott-continuous functions
(preserving joins, inducing frame-homs).

Directed-complete partial orders.

Start with a dcpo of states.
Events are Scott-open sets (a frame).
Morphisms are Scott-continuous functions
(preserving joins, inducing frame-homs).

Directed joins of random things are random things

Directed-complete partial orders.

Start with a dcpo of states.
Events are Scott-open sets (a frame).
Morphisms are Scott-continuous functions
(preserving joins, inducing frame-homs).

Directed joins of random things are random things
so dcpo is closed under random monad.

Directed-complete partial orders.

Start with a dcpo of states.
Events are Scott-open sets (a frame).
Morphisms are Scott-continuous functions
(preserving joins, inducing frame-homs).

Directed joins of random things are random things
so dcpo is closed under random monad.

dcpo is also closed under products, function spaces,
coinductive typess, ...

Directed-complete partial orders.

Start with a dcpo of states.
Events are Scott-open sets (a frame).
Morphisms are Scott-continuous functions
(preserving joins, inducing frame-homs).

Directed joins of random things are random things
so dcpo is closed under random monad.

dcpo is also closed under products, function spaces,
coinductive typess, ...

Continuous dcpos

...but dcpos don't have enough structure to be "domains".

Continuous dcpos

...but dcpos don't have enough structure to be "domains".

Continuous domains have more.

Continuous dcpos

...but dcpos don't have enough structure to be "domains".

Continuous domains have more.
CONT is closed under probability monad.

Continuous dcpos

...but dcpos don't have enough structure to be "domains".

Continuous domains have more.
CONT is closed under probability monad.
But NOT closed under function spaces.

Lattices I: states, randomness

Start with a lattice X

Lattices I: states, randomness

Start with a lattice X (e.g. real line).

Lattices I: states, randomness

Start with a lattice X (e.g. real line).
Let event space be the upper sets.

Lattices I: states, randomness

Start with a lattice X (e.g. real line).
Let event space be the upper sets.
To each valuation p, define a cpdf

$$p'(x) = p(\text{upper } x)$$

Dually, each cpdf d extends to a unique valuation

$$d'(\text{upper } x) = d(x)$$

Try again: (no event space)

Lattices I: states, randomness

Start with a lattice X (e.g. real line).
Let event space be the upper sets.
To each valuation p, define a cpdf

$$p'(x) = p(\text{upper } x)$$

Dually, each cpdf d extends to a unique valuation

$$d'(\text{upper } x) = d(x)$$

Try again: (no event space)
Morphisms are lattice homs.

Lattices I: states, randomness

Start with a lattice X (e.g. real line).
Let event space be the upper sets.
To each valuation p, define a cpdf

$$p'(x) = p(\text{upper } x)$$

Dually, each cpdf d extends to a unique valuation

$$d'(\text{upper } x) = d(x)$$

Try again: (no event space)
Morphisms are lattice homs.
Random states are cpdfs, but...

NO random monad

A space of random lattice elements need not itself be a lattice.

NO random monad

A space of random lattice elements need not itself be a lattice.

Hence Rand$\circ$Rand need not exist.

NO random monad

A space of random lattice elements need not itself be a lattice.

Hence Rand∘Rand need not exist.

Example

the square lattice $\bot \sqsubseteq \mathrm{tr}, \mathrm{fa} \sqsubseteq \top$,

NO random monad

A space of random lattice elements need not itself be a lattice.

Hence Rand$\circ$Rand need not exist.

Example

the square lattice $\perp \sqsubseteq \text{tr}, \text{fa} \sqsubseteq \top$,

$$\perp + \text{tr} \mid \perp + \text{fa} \sqsubseteq \perp + \top, \ \text{tr} + \text{fa}$$

NO random monad

A space of random lattice elements need not itself be a lattice.

Hence Rand$\circ$Rand need not exist.

Example

the square lattice $\bot \sqsubseteq \mathrm{tr}, \mathrm{fa} \sqsubseteq \top$,

$\bot + \mathrm{tr} \mid \bot + \mathrm{fa} \sqsubseteq \bot + \top, \; \mathrm{tr} + \mathrm{fa}$

no unique minimal upper bound

The max of two cdfs may lead to negative densities.

Abstract probability algebras

Start with a dcpo with $\perp$.

Abstract probability algebras

Start with a dcpo with $\perp$.

Generate initial "R-algebra" with binary mixing $x + y$

Abstract probability algebras

Start with a dcpo with $\perp$.
Generate initial "R-algebra" with binary mixing $x + y$ subject to monotonicity and

$$x + x = x \qquad \textit{idempotence}$$
$$x + y = y + x \qquad \textit{commutativity}$$

Abstract probability algebras

Start with a dcpo with $\bot$.

Generate initial "R-algebra" with binary mixing $x + y$ subject to monotonicity and

$x + x = x$	*idempotence*
$x + y = y + x$	*commutativity*
$(\omega + x) + (y + z) = (\omega + z) + (y + x)$	*associativity*

Abstract probability algebras

Start with a dcpo with $\bot$.
Generate initial "R-algebra" with binary mixing $x + y$ subject to monotonicity and

$x + x = x$	*idempotence*
$x + y = y + x$	*commutativity*
$(\omega + x) + (y + z) = (\omega + z) + (y + x)$	*associativity*

- equivalent to arbitrary real mixing
- equivalent to valuations

Abstract probability algebras

Start with a dcpo with $\perp$.
Generate initial "R-algebra" with binary mixing $x + y$ subject to monotonicity and

$x + x = x$	*idempotence*
$x + y = y + x$	*commutativity*
$(\omega + x) + (y + z) = (\omega + z) + (y + x)$	*associativity*

- equivalent to arbitrary real mixing
- equivalent to valuations

Compare with initial join-semilattice ("J-algebra")

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

Probability and lattices

Problem the join/meet of two random things may not be a random thing.

Probability and lattices

Problem: the join/meet of two random things may not be a random thing.

R-algebra models randomness.

Probability and lattices

Problem the join/meet of two random things may not be a random thing.

R-algebra models randomness.
J-algebra models parallelism.

Probability and lattices

Problem the join/meet of two random things may not be a random thing.

R-algebra models randomness.
J-algebra models parallelism.
JR-algebra with distributivity.

$$(x + y) \mid z = (x \mid z) + (y \mid z)$$

Probability and lattices

Problem the join/meet of two random things may not be a random thing.

R-algebra models randomness.
J-algebra models parallelism.
JR-algebra with distributivity.

$$(x + y) \mid z = (x \mid z) + (y \mid z)$$

models parallelism with randomness,

Probability and lattices

Problem the join/meet of two random things may not be a random thing.

R-algebra models randomness.
J-algebra models parallelism.
JR-algebra with distributivity.

$$(x + y) \mid z = (x \mid z) + (y \mid z)$$

models parallelism with randomness,
allows random normal form,

Probability and lattices

Problem the join/meet of two random things may not be a random thing.

R-algebra models randomness.
J-algebra models parallelism.
JR-algebra with distributivity.

$$(x + y) \mid z = (x \mid z) + (y \mid z)$$

models parallelism with randomness,
allows random normal form,
sampling semantics

Probability and lattices

Problem the join/meet of two random things may not be a random thing.

R-algebra models randomness.
J-algebra models parallelism.
JR-algebra with distributivity.

$$(x + y) \mid z = (x \mid z) + (y \mid z)$$

models parallelism with randomness,
allows random normal form,
sampling semantics
JR-algebra models... nothing nice,

Probability and lattices

Problem the join/meet of two random things may not be a random thing.

R-algebra models randomness.
J-algebra models parallelism.
JR-algebra with distributivity.

$$(x + y) \mid z = (x \mid z) + (y \mid z)$$

models parallelism with randomness,
allows random normal form,
sampling semantics
JR-algebra models... nothing nice,
NO random normal form

Probability and lattices

Problem the join/meet of two random things may not be a random thing.

R-algebra models randomness.
J-algebra models parallelism.
JR-algebra with distributivity.

$$(x + y) \mid z = (x \mid z) + (y \mid z)$$

models parallelism with randomness,
allows random normal form,
sampling semantics
JR-algebra models... nothing nice,
NO random normal form
Next time: can JR-algebras be made to work?

Summary and prospects

(again)
We started with finite sets,

Summary and prospects

(again)
We started with finite sets,
generalized to probability measures,

Summary and prospects

(again)
We started with finite sets,
generalized to probability measures, then
weakened the event language,

Summary and prospects

(again)
We started with finite sets,
generalized to probability measures, then
weakened the event language,
added structure among points,

Summary and prospects

(again)
We started with finite sets,
generalized to probability measures, then
weakened the event language,
added structure among points, and
ended with fully algebraic approaches.