Probability, Nondeterminism and Concurrency

Daniele Varacca

PhD Defence

BRICS

Aarhus, November 24th 2003

Road Map

- Motivation
- Probability and Nondeterminism
- Probability and Concurrency

Road Map

- Motivation
  - Understanding the nature of computation
  - Mathematical semantics
  - Probability, Nondeterminism and Concurrency
- Probability and Nondeterminism
- Probability and Concurrency

The Nature of Computation

It's a wild world out there:

- complex needs: precision, speed, security
- complex tools: languages, protocols, networks, channels

What do computers do, really?

Semantics

Semantics is a tool to find our way

- programs are associated to mathematical objects (the meaning)
- mathematical objects are parts of structures (the models)

Goal: study the models to understand the programs

Semantics

Requirements for the models

- models should be complex enough to reflect interesting properties of real computation
- models should be simple enough to be studied

Right balance between abstraction and complexity

The meaning of a complex system should be built using the meanings of simpler parts of the system (compositionality)

Concurrency

Computers running independently, sharing some components, and communicating between them

Models must be able to represent

- concurrency (parallelism)
- shared components (synchronisation - conflict)
- communication (channels, messages)

Probability

A model is probabilistic when

- different alternatives are represented
- every alternative is assigned a probability

We need probability

- to model failures and unreliability
- to produce efficient algorithms
- to enhance security

Nondeterminism

A model is nondeterministic when

- different alternatives are represented
- the way one alternative is chosen over the others is not specified

We need nondeterminism

- to abstract away from details
- to achieve compositionality
- to model concurrency (sometimes)

My work

The rest of the talk will present

- a mathematical model combining probability and nondeterminism
- a mathematical model combining probability and concurrency

Road Map

- Motivation
- Probability and Nondeterminism
- Probability and Concurrency

Road Map

- Motivation
- Probability and Nondeterminism
    - Categorical Semantics and monads
    - Nondeterministic monad, probabilistic monad
    - Combining the two monads
    - The *indexed* probabilistic monad
- Probability and Concurrency

Categorical semantics

Idea: programs take in input values of type X and return values of type Y

Programs are represented as arrows

$$X \xrightarrow{p} Y$$

Programs compose:

$$\frac{X \xrightarrow{p} Y, Y \xrightarrow{q} Z}{X \xrightarrow{p;q} Z}$$

Example: sets and functions

Computational effects

Programs can have effects

- nontermination
- nondeterminism
- probability
- permanent state

Idea: programs take in input values of type X and return generalized values of type Y

Monads

A monad consists of

- the operator T: if X represents values, $T(X)$ represents generalized values
- the unit $X \xrightarrow{\eta_X} T(X)$: values are special kinds of generalized values
- the extension $(-)^\dagger$: if $X \xrightarrow{f} T(Y)$, then $T(X) \xrightarrow{f^\dagger} T(Y)$

satisfying some axioms

Monads

- The extension models sequential composition:

$$\frac{X \xrightarrow{f} T(Y), \ Y \xrightarrow{g} T(Z)}{X \xrightarrow{f} T(Y) \xrightarrow{g^\dagger} T(Z)}$$

- Specific effects have also specific operations

Monads: examples

The nondeterministic monad

- the operator is the powerset $P(X)$
- the unit: $x \mapsto \{x\}$
- the extension: if $X \xrightarrow{f} P(Y)$, and if $A \in P(X)$, then
$$f^{\dagger}(A) = \bigcup_{x \in A} f(x)$$

A program returns a set of values

Choice is modelled by union of sets

Monads: examples

The probabilistic monad

- the valuation operator $V(X)$: functions $f : X \to \overline{\mathbb{R}^+}$
- the unit $x \mapsto \delta_x$
- the extension is obtained by weighted average

A program returns a valuation over values
Probabilistic choice $x +_p y$ is modelled by convex combination

$$x +_p x = x$$

Combining monads

We want a program return a set of valuations.

- the operator is the the composition of the operators $P(V(X))$
- the unit is the composition of the units
- the extension

Combining monads

We want a program return a set of valuations.

- the operator is the the composition of the operators $P(V(X))$
- the unit is the composition of the units
- the extension does not work!

We need to change something

Solutions

- Solution 1 (Tix, Mislove): change the sets
  Use convex sets of valuations:
  if $\nu, \xi \in A$, then $\nu +_p \xi \in A$

  The operator $P_{convex} \circ V$ forms a monad

Solutions

- Solution 1 (Tix, Mislove): change the sets
  Use convex sets of valuations:
  if $\nu, \xi \in A$, then $\nu +_p \xi \in A$

  The operator $P_{convex} \circ V$ forms a monad
- Solution 2 (Varacca, Winskel): change the valuations
  Use indexed valuations

Indexed valuations

An indexed valuation over a set X is given by

- an indexing set I
- an indexing function $I \to X$
- a valuation over I

Notation

$$\nu = (x_i, p_i)_{i \in I}$$

Indexed valuations

Intuition:

- elements of X represent observations
- elements of I represent computations
- the indexing function associates computations to observations

The probability on observations is gotten from the computations

Like in a... random variable!

The monad

We have a monad:

- the operator is $IV(X) = \{$indexed valuations over $X\}$
- the unit: $x \mapsto (x, 1)_{* \in \{*\}}$
- the extension is obtained via disjoint union of indices

The probabilistic choice is also modelled by disjoint union of the indices

$$x +_p x \neq x$$

Composing the monads

We can compose IV and P and get a monad
We can now model a programming language with

- sequential composition
- nondeterministic choice
- probabilistic choice

Ordering

Which order on indexed valuations?
Idea: the equation $x +_p x = x$ is weakened to
$x +_p x \geq x$

- Morally: the more indices, the better
- This combines well with the Hoare order on sets
  $A \cup B \geq A$
- $\rightsquigarrow$ domain theory

Other issues

More in the thesis

- equational theories
- other orderings
- relation with continuous valuations
- the problem of the concrete characterization
- many technical details

Papers

- *The Powerdomain of Indexed Valuations* - LICS 2002
- a longer BRICS report with the same title

Related work

- CSP group in Oxford
- Tix' thesis and Mislove's paper
- Plotkin, Power, Hyland on monads and operations

Future work

- concrete characterization
- presheaf models?

Road Map

- Motivation
- Probability and Nondeterminism
- Probability and Concurrency

Road Map

- Motivation
- Probability and Nondeterminism
- Probability and Concurrency
  - Event structures
  - Confusion
  - Probabilistic event structures
  - ...and domains
  - further issues

Event Structures

Triple $\mathcal{E} = \langle E, \leq, \# \rangle$

E is a set of events

$\leq$ is the causal dependency relation

$\#$ is the conflict relation

- $\langle E, \leq \rangle$ is a partial order
- for every $e \in E$, $e \downarrow$ is finite
- $\#$ is irreflexive and symmetric
- $\#$ is "hereditary": $e_1 \# e$ and $e_1 \leq e_2$ implies $e_2 \# e$

Configurations

A notion of a run of an event structure

A configuration is a set x of events

- justified: $e \in x, e' \leq e \implies e' \in x$
- conflict-free: $e, e' \in x \implies \neg e \# e'$

Examples:

$$[e] := \{e' \,|\, e' \leq e\}$$

$$[e) := [e] \setminus \{e\}$$

If $[e) \subseteq x$, e is enabled at x

The set of configurations is $\mathcal{L}(\mathcal{E})$

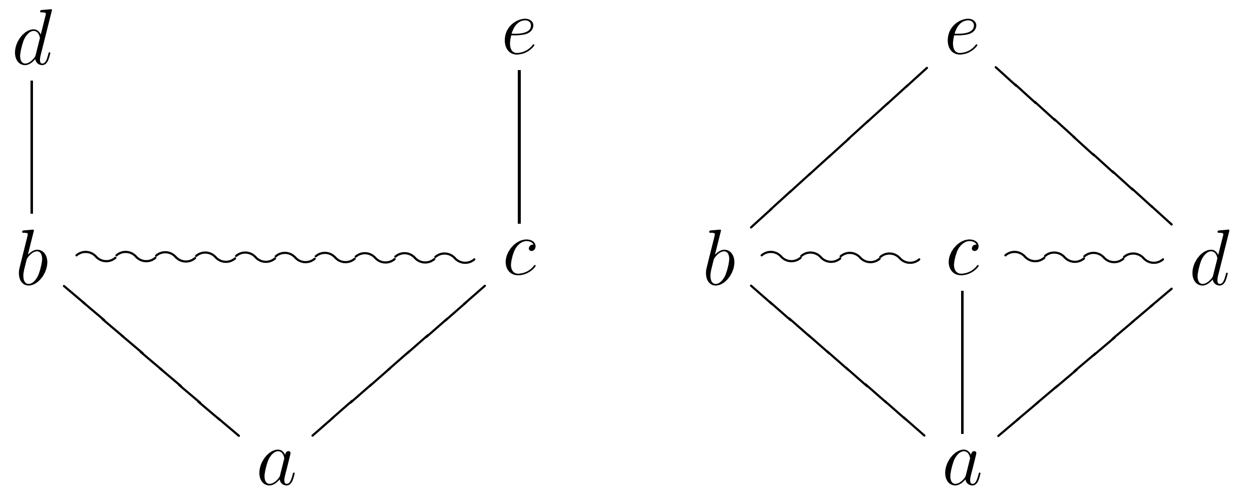
Immediate Conflict

$e \#_\mu e'$ iff

- $e \# e'$ and
- $[e] \cup [e'), [e) \cup [e']$ are configurations

Two configurations x, x' are compatible if $x \cup x'$ is a configuration

Event Structures: Examples



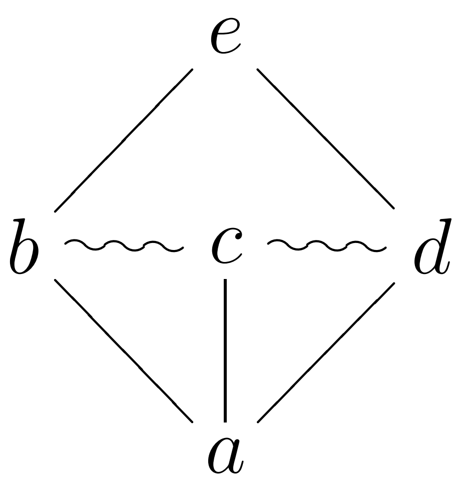
(A) (B)

A configuration of A: $\{a, b, d\}$

A configuration of B: $\{a, b, d, e\}$

Probabilistic choice?

We want to resolve the conflicts by flipping a coin



How do we resolve the conflict between b, c, d?

Confusion

An event structure is confusion-free when

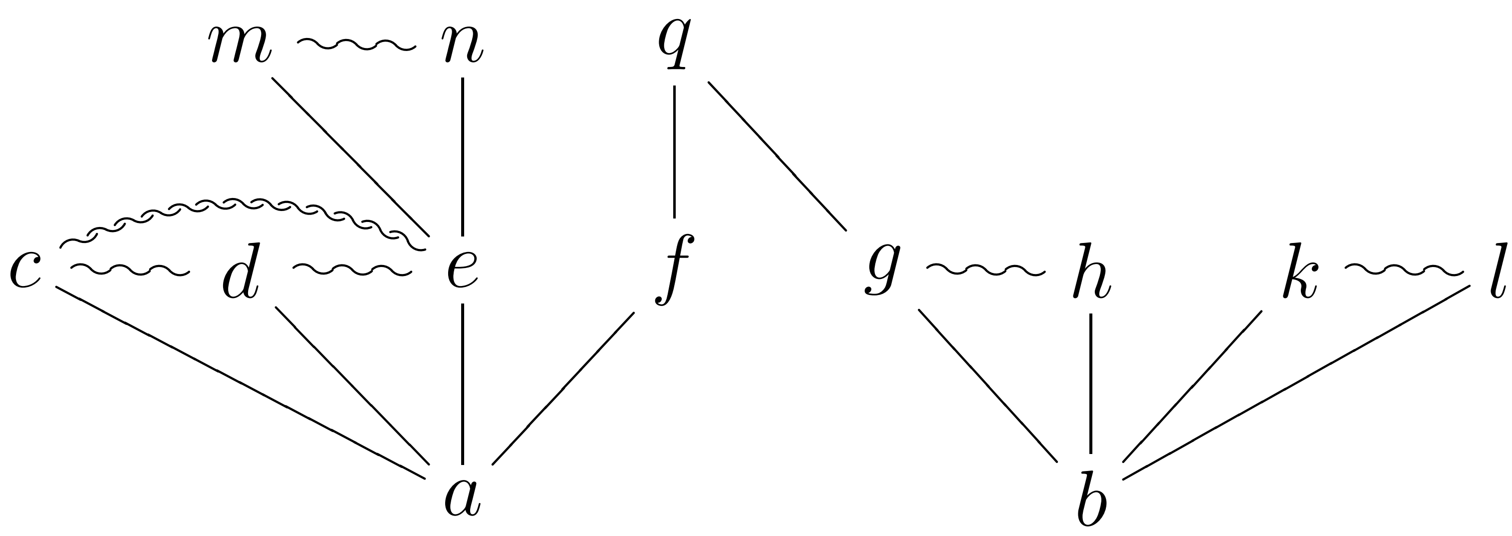
- $\#_\mu \cup 1_E$ is an equivalence
- $e \#_\mu e' \Longrightarrow [e) = [e')$

The equivalence classes are the cells

A cell c is enabled at x if one (and therefore all) of its events is enabled at x
A cell c is filled by x if there is $e \in c \cap x$

A cell c is accessible at x if c is enabled at x, but c is not filled by x

Confusion-freeness: Example



Configurations: $\{a, b, f, g, q\}, \{a, b, e, n\}$

Valuations on Event Structures

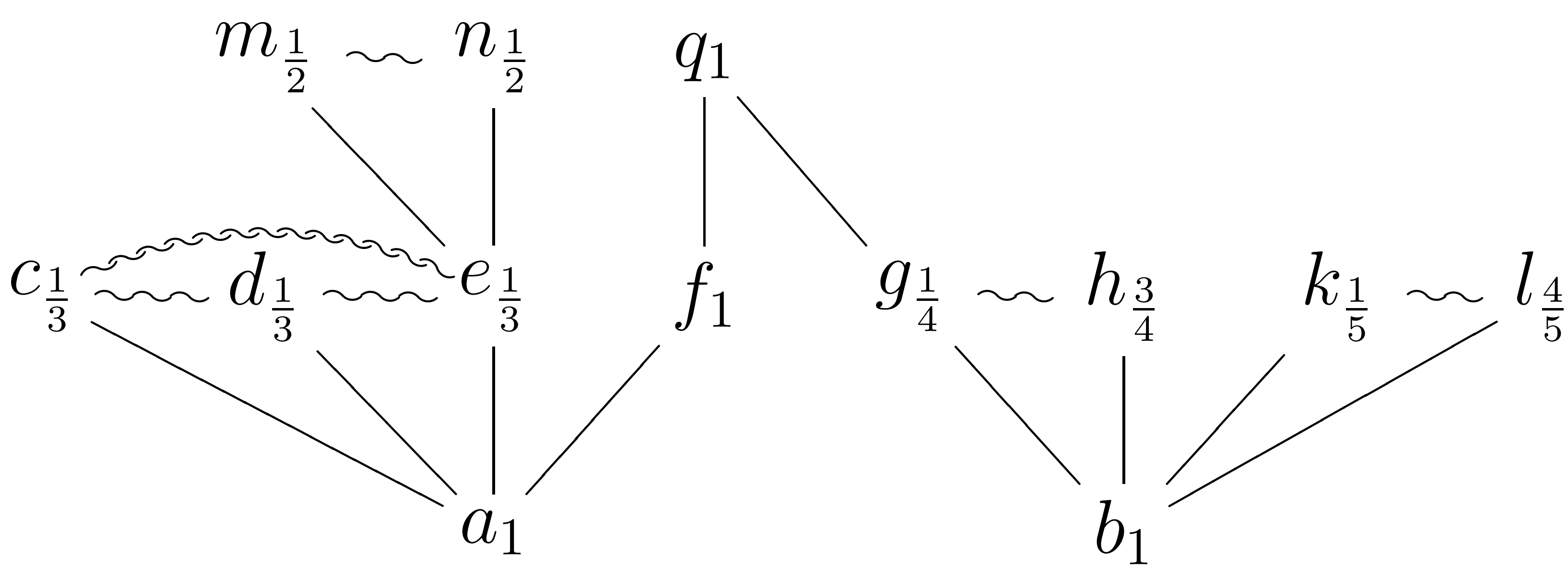
A valuation on $\mathcal{E}$ is a function $p : E \to [0, 1]$ such that for every cell c

$$\sum_{e \in c} p(e) = 1$$

We define a function $v_p : \mathcal{L}(\mathcal{E}) \to [0, 1]$

$$v_p(x) = \prod_{e \in x} p(e)$$

Valuations: Example



Configurations:

$$v_p(\{a,b,f,g,q\}) = \frac{1}{4}, v_p(\{a,b,e,n\}) = \frac{1}{6}$$

Test

A set C of configurations is test when

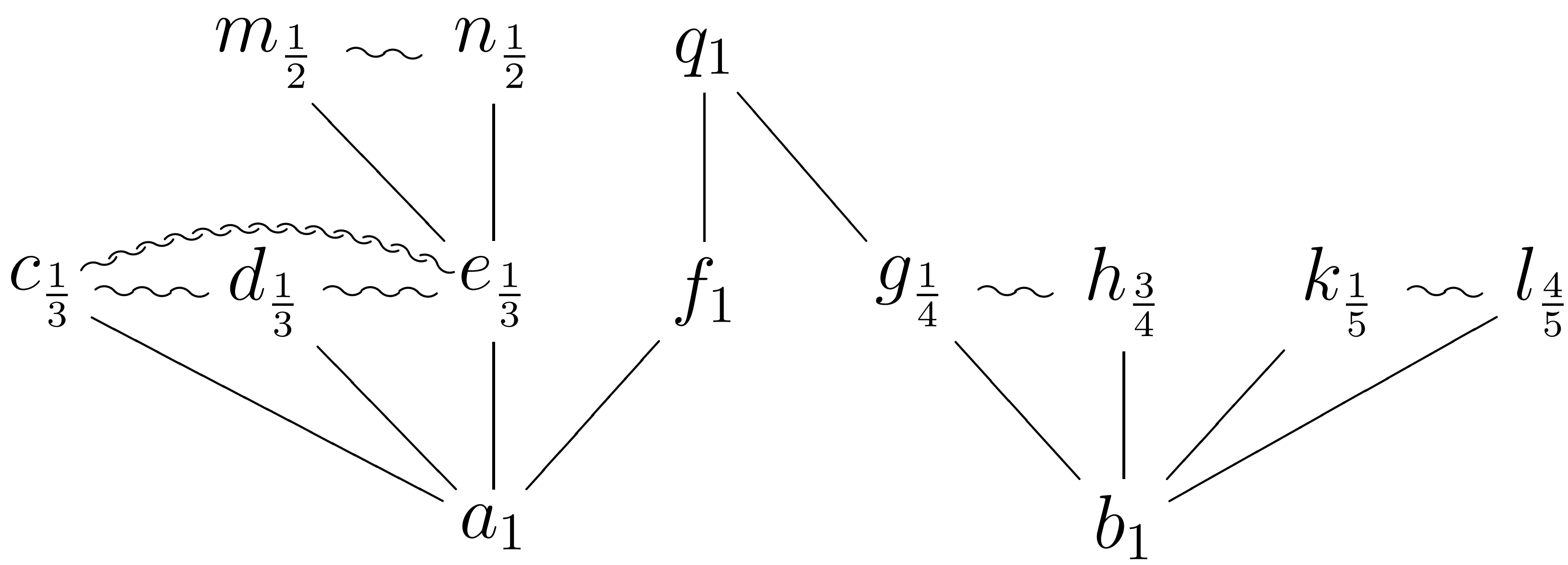
- If $x, x' \in C$, then x, x' are not compatible (incompatibility)
- Every configuration of $\mathcal{E}$ is compatible with some $x \in C$ (maximality)

Tests represent probabilistic runs
Runs can be extended:

$C \leq C'$ if for every $x \in C$ there is $x' \in C'$, $x \subseteq x'$ and

for every $x' \in C'$ there is $x \in C$, $x \subseteq x'$

Tests: Example

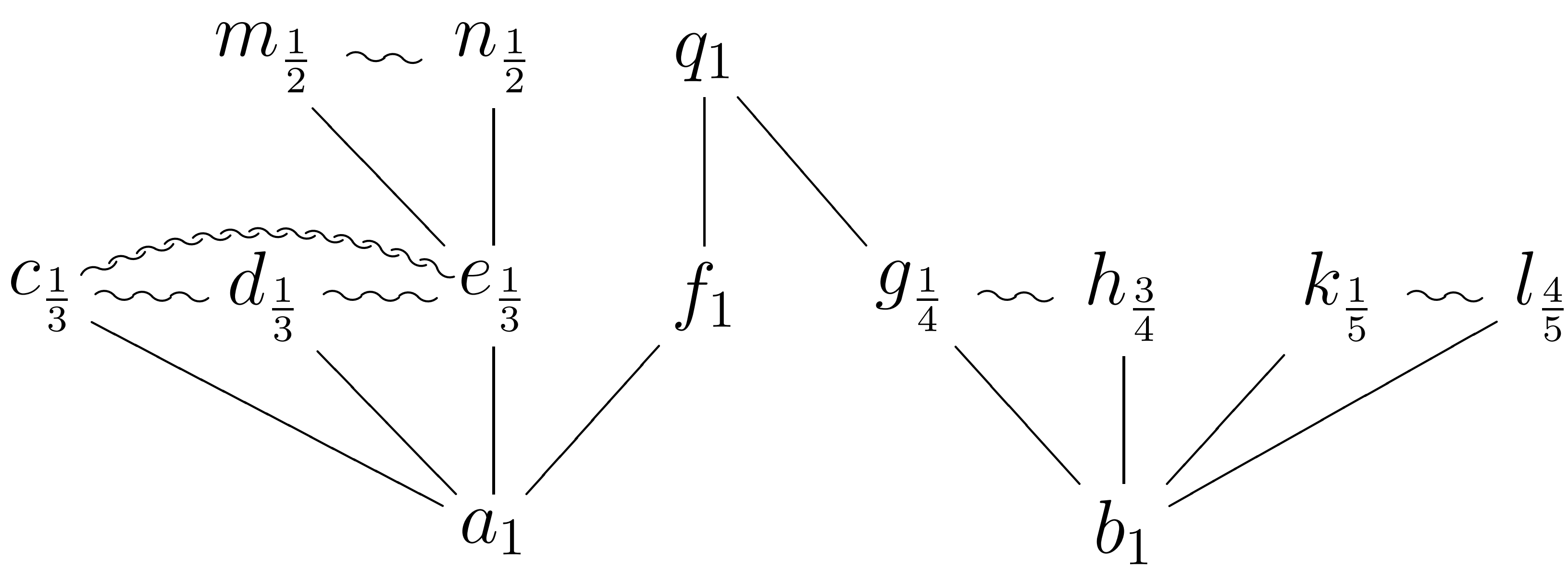


Two tests

$$\{a, b, f, g, q\}, \{a, b, f, h, k\}, \{a, b, f, h, l\}$$

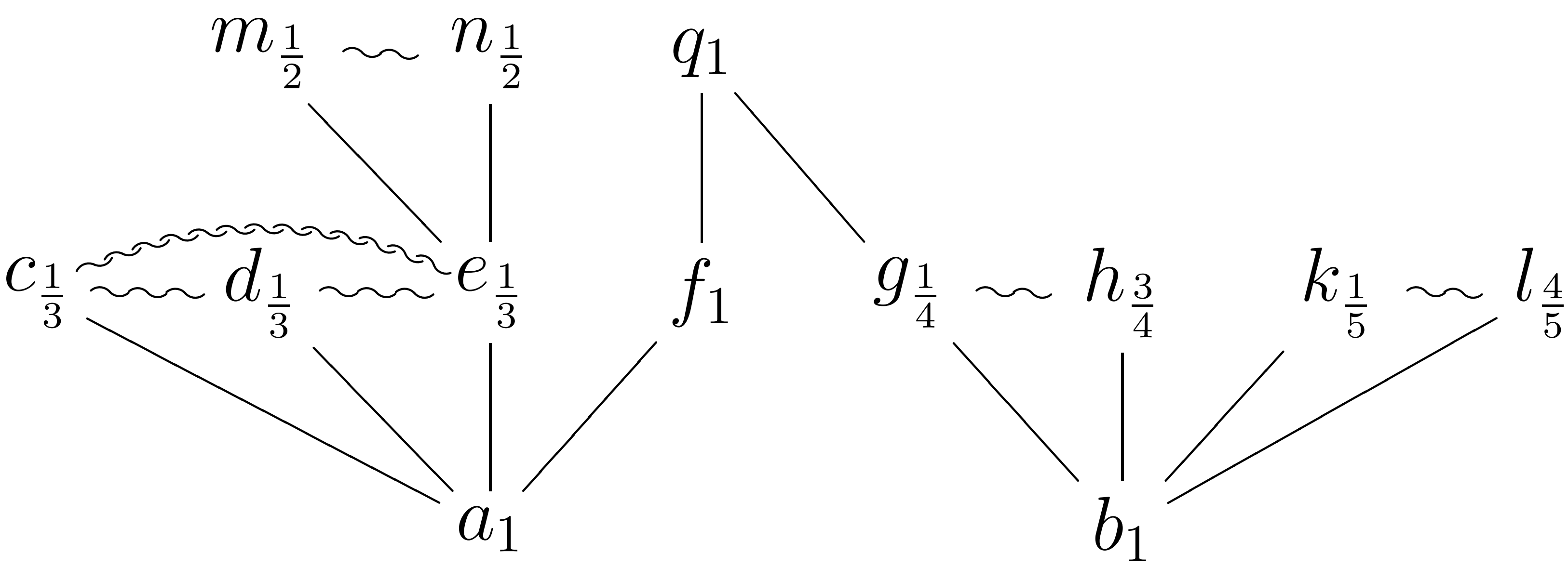
$$\{a, b, e, n\}, \{a, b, e, m\}, \{a, b, d, f\}, \{a, b, c, g\}, \{a, b, c, h\}$$

Tests: Example



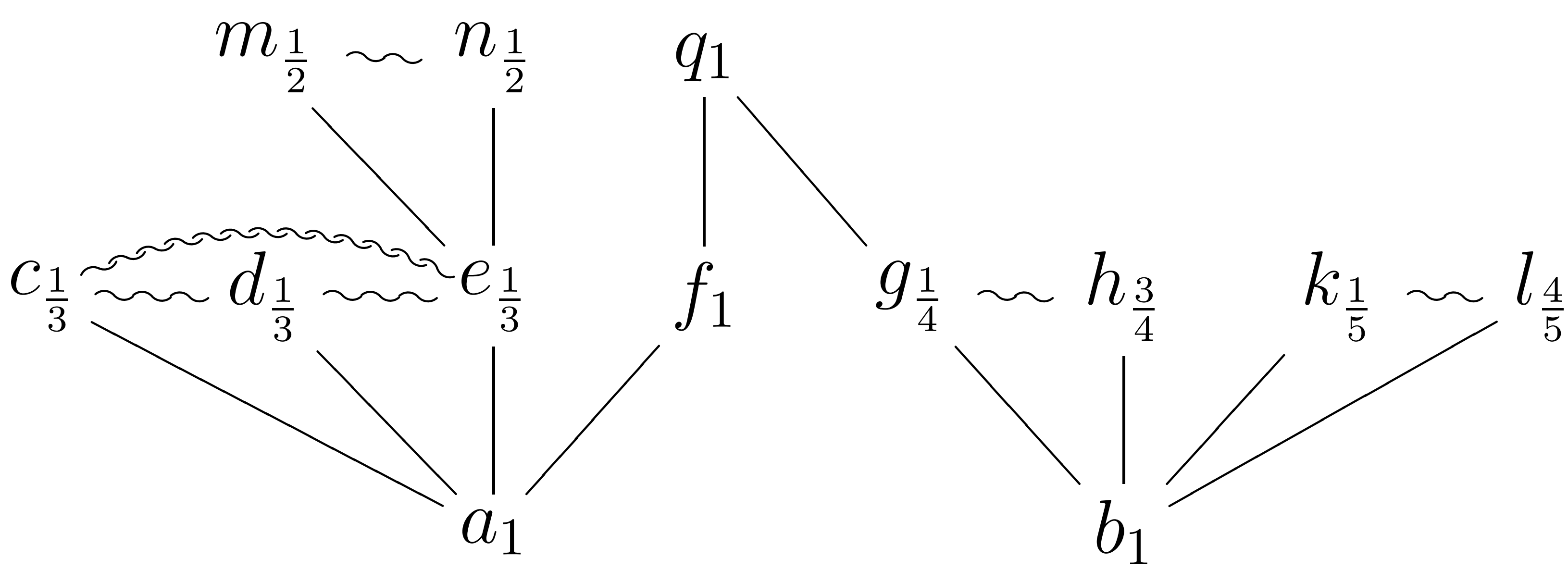
$$\emptyset \rightarrow \{a, b\}$$

Tests: Example



$$\{a, b\} \rightarrow \{a, b, c\}, \{a, b, d\}, \{a, b, e\}$$

Tests: Example

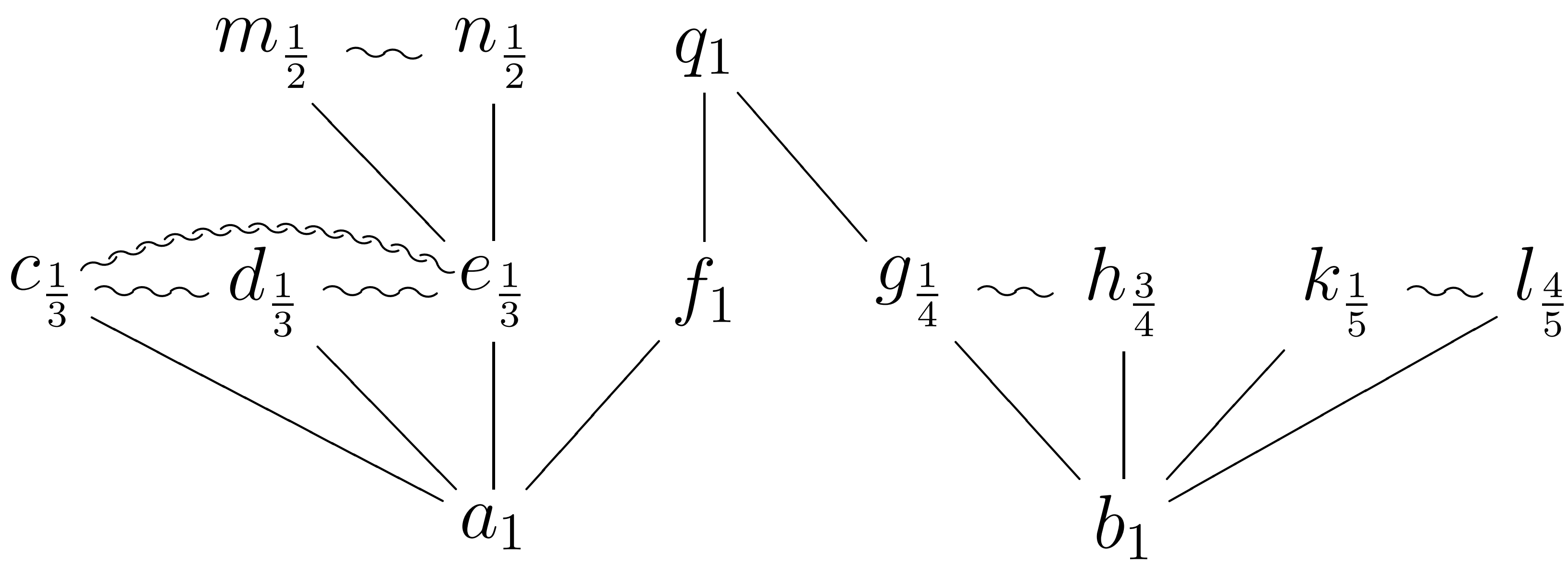


$$\{a, b, c\} \rightarrow \{a, b, c, g\}, \{a, b, c, h\}$$

$$\{a, b, d\} \rightarrow \{a, b, d, f\}$$

$$\{a, b, e\} \rightarrow \{a, b, e, m\}, \{a, b, e, n\}$$

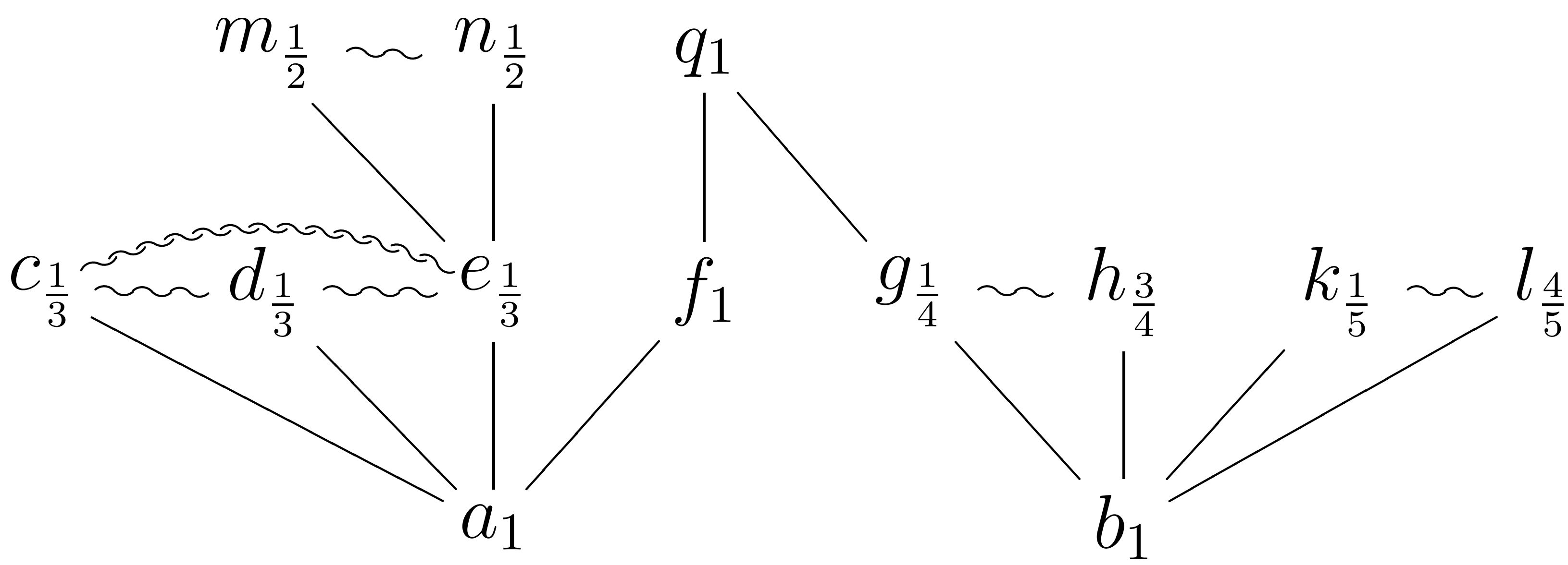
Tests: Example



All together

$$\{a,b,e,n\},\{a,b,e,m\},\{a,b,d,f\},\{a,b,c,g\},\{a,b,c,h\}$$

Tests: Example



$$v_p(\{a,b,e,n\}) = \frac{1}{6}, v_p(\{a,b,e,m\}) = \frac{1}{6}$$
$$v_p(\{a,b,d,f\}) = \frac{1}{3}, v_p(\{a,b,c,g\}) = \frac{1}{12}$$
$$v_p(\{a,b,c,h\}) = \frac{1}{4}$$

Tests are runs

Proposition If C is a test, then

$$\sum_{x \in C} v_p(x) = 1$$

Nondeterminism

There is a nondeterministic choice as to which cell to fire.

The two tests

$$\begin{array}{rcl} C & := & \{a,b,f,g,q\},\{a,b,f,h,k\},\{a,b,f,h,l\} \\ C' & := & \{a,b,e,n\},\{a,b,e,m\}, \\ & & \{a,b,d,f\},\{a,b,c,g\},\{a,b,c,h\} \end{array}$$

are incomparable

$$C \not\leq C',\ C' \not\leq C$$

However

No Nondeterminism

Theorem

If C, C' are tests, then there exists a test C'', with $C, C' \leq C''$

Morally: the nondeterministic branching does not matter

Event Structures and Domains

The set $\mathcal{L}(\mathcal{E})$ of configurations ordered by inclusion

- is an algebraic DCPO
- its compact elements are the finite configurations

The sets of the form $x \uparrow$ are Scott-open when x is finite

Continuous Valuations

A continuous valuation is a function assigning a weight to all Scott-open sets, satisfying some (reasonable) axioms

Theorem
For every valuation p on an event structure $\mathcal{E}$ there is a unique continuous valuation ν_p on $\mathcal{L}(\mathcal{E})$ such that, for every finite configuration x:

$$\nu_p(x\uparrow) = v_p(x) = \prod_{e \in x} p(e)$$

Other issues

More in the thesis

- beyond stochastic independence
- partial probabilities
- categorical observations
- probabilistic Mazurkiewicz equivalence
- technical details

Papers

- *Probabilistic Petri Nets and Mazurkiewicz Equivalence*, with Mogens Nielsen - Draft
- *Probabilistic Petri Nets, Event Structures and Domains*, with Hagen Völzer and Glynn Winskel - In preparation

Related work

- Katoen's Probabilistic Event Structures
- Völzer's thesis
- Benveniste, Fabre, Haar, Abbes at Rennes

Future (present?) work

- relating the related work
- beyond confusion freeness
- “concrete”applications
- continuous probabilities
- bisimulation, logics, verification...

Men det er en anden historie,
og det må vente til en anden gang

The question was: