Measuring Probabilistic Contracts

IFIP WG2.1 meeting #75

Montevideo, Uruguay, February 2017

J.N. Oliveira

INESC TEC & University of Minho

(Grant FP7-ICT 619606)

Context

TRUST

Trustworthy Software Design with Alloy

In this project we are extending **Alloy** to make it a more **comprehensive** modelling tool for trustworthy software design.

One direction is to be able to quantify **faulty behaviour** in software models, in a **probabilistic** way.

We thought of extending Alloy's underlying Boolean matrices to **stochastic** matrices and of adapting the notion of **contract validity** accordingly.

This talk will present and discuss a **linear algebra** approach to *"measuring"* contract validity.

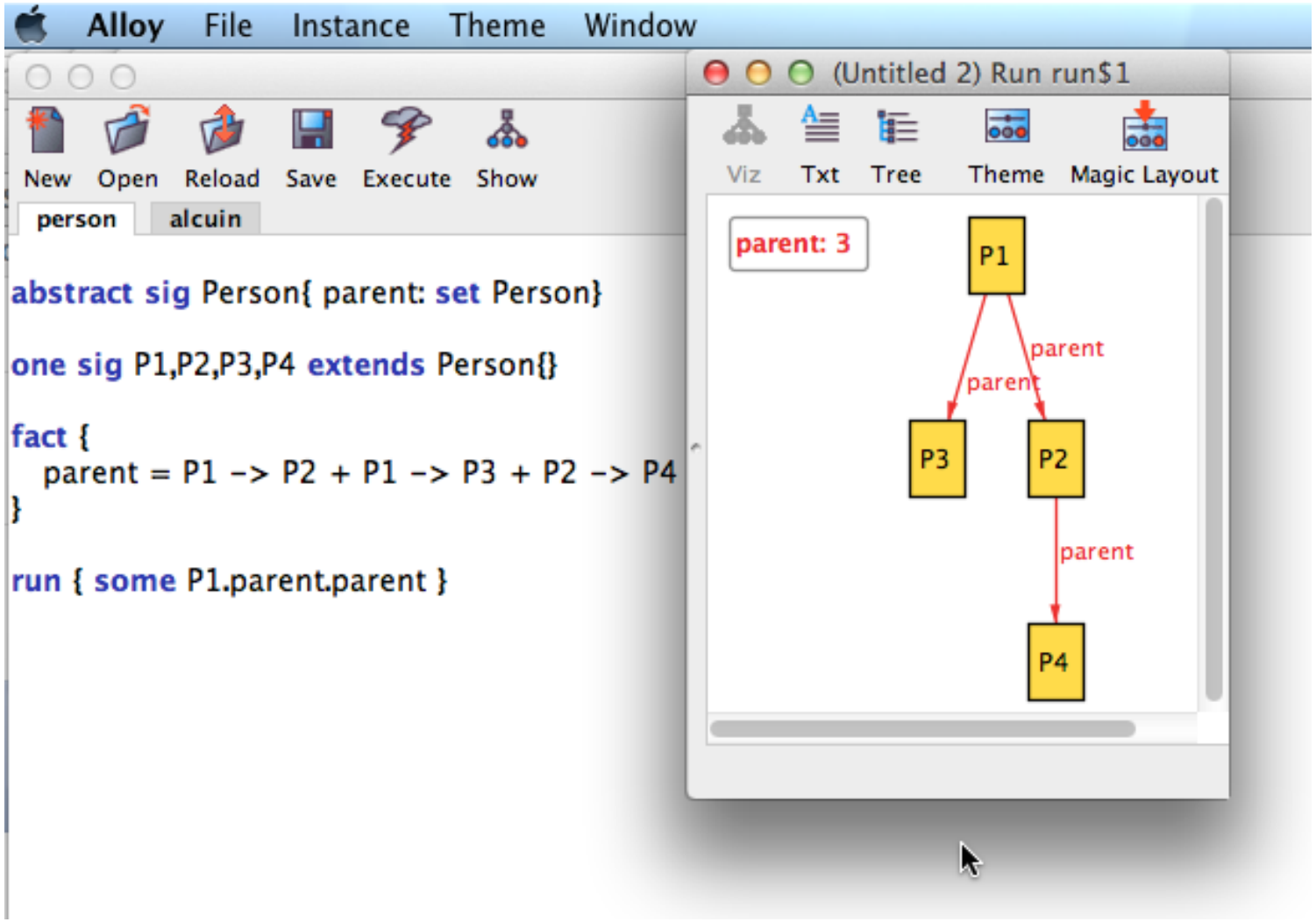
Alloy

Relational **composition**:

- The Swiss army knife of Alloy
- It subsumes **function application** and "**field selection**"
- Encourages a **navigational** (point-free) style based on pattern $x.(R.S)$.
- Example:

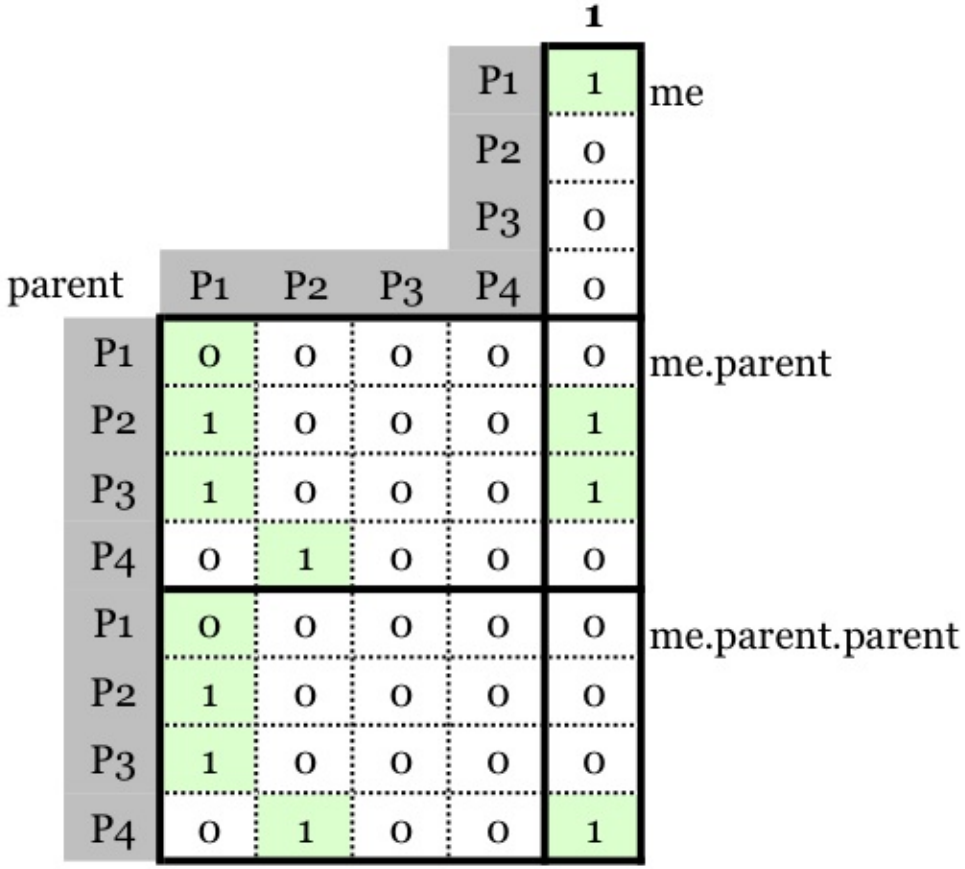
$$Person = \{(P1),(P2),(P3),(P4)\}$$
$$parent = \{(P1,P2),(P1,P3),(P2,P4)\}$$
$$me = \{(P1)\}$$
$$me.parent = \{(P2),(P3)\}$$
$$me.parent.parent = \{(P4)\}$$
$$Person.parent = \{(P2),(P3),(P4)\}$$

Alloy


Alloy File Instance Theme Window
New Open Reload Save Execute Show
person alcuin
abstract sig Person{ parent: set Person}
one sig P1,P2,P3,P4 extends Person{}
fact {
parent = P1 -> P2 + P1 -> P3 + P2 -> P4
}
run { some P1.parent.parent }
(Untitled 2) Run run$1
Viz Txt Tree Theme Magic Layout
parent: 3
P1
parent
parent
P3
P2
parent
P4

Alloy

Relations are Boolean matrices:



Note how *me*, *me.parent* etc are all at most $\mathit{Person} \xleftarrow{!^\circ} 1$, where $! \_ = 1$ (the everywhere-1 function).

Functions are Boolean matrices

A relation $B \xleftarrow{V} A$ is said to be a **vector** if either A or B are the singleton type 1.

Relation $1 \xleftarrow{V} A$ is said to be a **row**-vector; clearly, $V \subseteq !$

Relation $B \xleftarrow{V} 1$ is said to be a **column**-vector; clearly, $V \subseteq !^{\circ}$

Functions are Boolean matrices f such that

$$! \cdot f = ! \tag{1}$$

for instance $\begin{bmatrix} 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$

NB: mind the two **polymorphic** copies of $! : A \to 1$ in (1).

Evolution

Alloy's "dot-join" $r \cdot s$ generalizes **function** composition

$$(f \cdot g)\, x \quad = \quad f\, (g\, x)$$

to **relation** composition,

$$y\, (R \cdot S)\, x \quad \Leftrightarrow \quad \langle \exists\, z \;::\; y\, R\, z \wedge z\, S\, x \rangle$$

itself generalizable to **matrix** composition ("multiplication")

$$y\, (M \cdot N)\, x \quad = \quad \langle \sum z \;::\; (y\, M\, z) \times (z\, N\, x) \rangle$$

where the infix $y\ M\ z$ linking to relational notation is intentional.

In my view, any modelling tool should live peacefully with this *function* $\rightarrow$ *relation* $\rightarrow$ *matrix* evolution in **expressiveness**.

The evolution

Determinism (*functions*):

- Functional programming (FP)
- Imperative programming (if restricted)

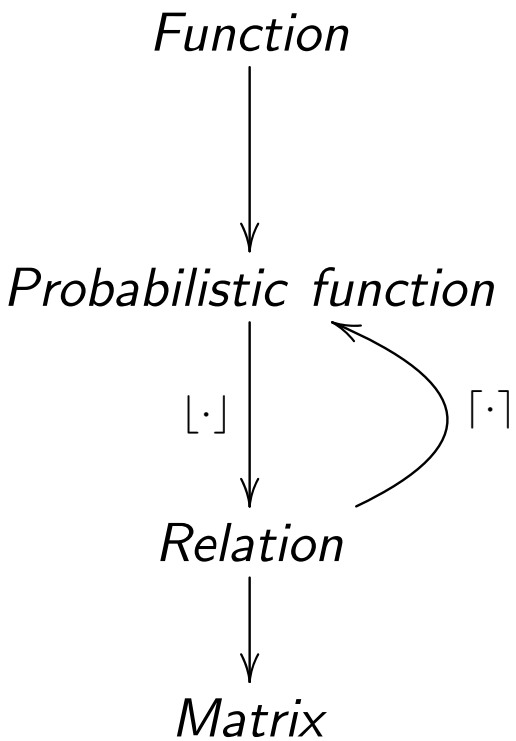
Non-determinism (*relations*):

- Logic programming
- Relational modelling

Probabilism (*matrices*):

- Probabilistic modelling
- Quantum programming

Probabilistic functions



$$f = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$g = \begin{bmatrix} 1 & 0 & 0.5 & 0.1 \\ 0 & 0.7 & 0 & 0.9 \\ 0 & 0.3 & 0.5 & 0 \end{bmatrix}$$

$$R = \lfloor g \rfloor = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Both f and g satisfy (1) ; moreover, $\lceil R \rceil = \begin{bmatrix} 1 & 0 & 0.5 & 0.5 \\ 0 & 0.5 & 0 & 0.5 \\ 0 & 0.5 & 0.5 & 0 \end{bmatrix}$

Probabilistic functions vs relations

Galois connection

$$\lfloor f \rfloor \subseteq R \Leftrightarrow f \leqslant \lceil R \rceil \tag{2}$$

such that

$$\lfloor \lceil R \rceil \rfloor = R$$

— that is, $\lceil \_ \rceil$ is **injective** and $\lfloor \_ \rfloor$ is **surjective**.

This enables us to regard the latter (supports) as a relational **abstract interpretation** of probabilistic functions (PF).

The preorder $(\leqslant)$ ranks PFs according to (lack of) uniformity.

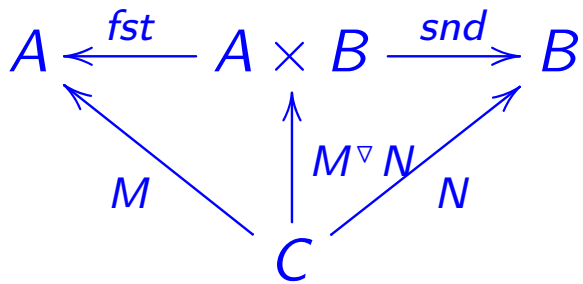
Monoidal categories

Every stage in the hierarchy forms a **monoidal category** whose **composition** has been given already, and whose **tensor** is based on **pairing**:

$$M \otimes N = (M \cdot fst) \triangledown (N \cdot snd) \tag{3}$$

Pairing is a **weak product** for PFs:

$$k = M \triangledown N \Rightarrow \begin{cases} fst \cdot k = M \\ snd \cdot k = N \end{cases}$$



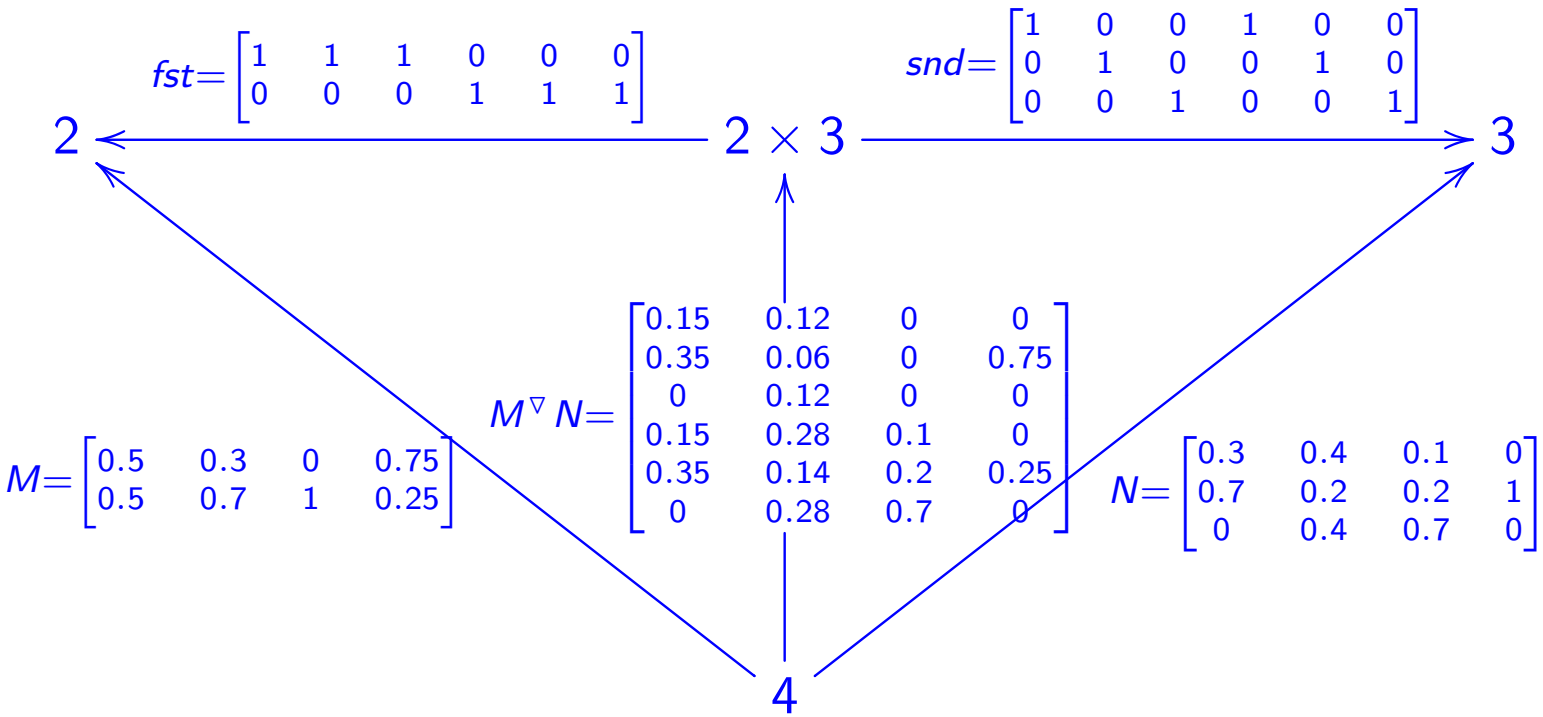
For **pure** functions this becomes a **full** categorial product.

Probabilistic pairing (Khatri-Rao)

In summary: weak product still grants the **cancellation** rule,

$$fst \cdot (M \triangledown N) = M \ \wedge \ snd \cdot (M \triangledown N) = N$$

cf. e.g.



Probabilistic pairing (entanglement)

... but **reconstruction**

$$X = (fst \cdot X) \triangledown (snd \cdot X)$$

doesn't hold in general, cf. e.g.

$$X : 2 \to 2 \times 3$$

$$X = \begin{bmatrix} 0 & 0.4 \\ 0.2 & 0 \\ 0.2 & 0.1 \\ 0.6 & 0.4 \\ 0 & 0 \\ 0 & 0.1 \end{bmatrix} \qquad (fst \cdot X) \triangledown (snd \cdot X) = \begin{bmatrix} 0.24 & 0.4 \\ 0.08 & 0 \\ 0.08 & 0.1 \\ 0.36 & 0.4 \\ 0.12 & 0 \\ 0.12 & 0.1 \end{bmatrix}$$

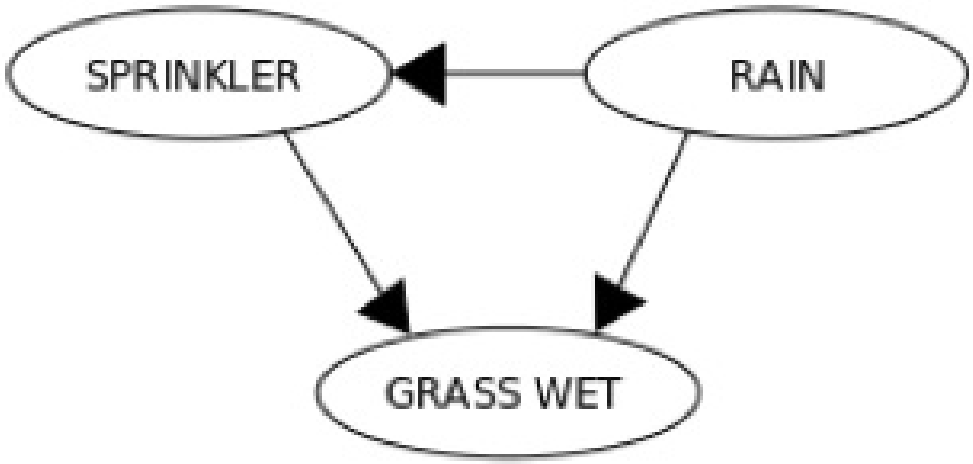
X is not recoverable from its projections: Khatri-Rao not **surjective**.

(**Entangled** distributions on pairs **fall outside** the range of the Khatri-Rao product.)

Illustration

Example adapted from

[`https://en.wikipedia.org/wiki/Bayesian_network`]



Control a **sprinkler** to wet the **grass** in case it does not **rain**.

Functional (deterministic) model

$S = R = G = 2$

$sprinkler : R \to S$
$sprinkler\ r = \neg\ r$

$grass : S \times R \to G$
$grass\ (s, r) = s \vee r$

$rain \in \{0, 1\}$

$$
\begin{array}{c}
G \times (S \times R) \\
\uparrow{\scriptstyle grass \triangledown id} \\
S \times R \\
\uparrow{\scriptstyle sprinkler \triangledown id} \\
R \\
\uparrow{\scriptstyle rain} \\
1
\end{array}
$$

Grass always wet:

$$grass\ (sprinkler\ r, r) = \neg\ r \vee r = \mathrm{T}$$

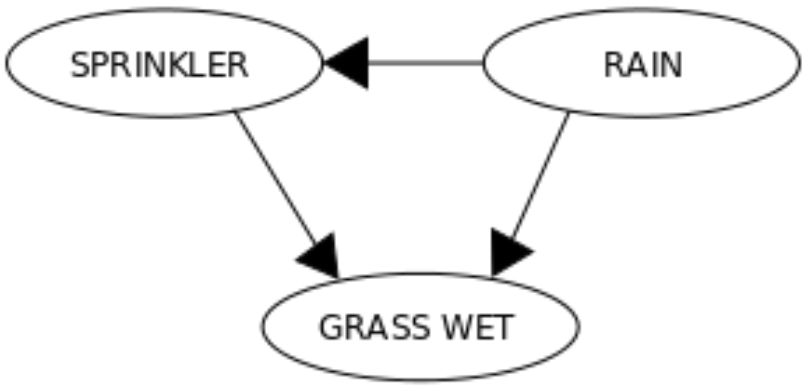
Altogether, two possible states $\{(1, (1, 0)), (1, (0, 1))\}$ of type:

$$G \times (S \times R) \xleftarrow{state} 1 \ = (grass \triangledown id) \cdot (sprinkler \triangledown id) \cdot rain$$

Bayesian networks

Previous model is not realistic — the picture actually found on Wikipedia is:

	SPRINKLER	
RAIN	T	F
F	0.4	0.6
T	0.01	0.99



	RAIN	
T	F	
0.2	0.8	

		GRASS WET	
SPRINKLER	RAIN	T	F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

Bayesian network (probabilistic model)

Let

$$S = R = G = 2$$

$$S \xleftarrow{sprinkler} R = \begin{bmatrix} 0.60 & 0.99 \\ 0.40 & 0.01 \end{bmatrix}$$

$$R \xleftarrow{rain} 1 = \begin{bmatrix} 0.80 \\ 0.20 \end{bmatrix}$$

$$G \xleftarrow{grass} S \times R = \begin{bmatrix} 1.00 & 0.20 & 0.10 & 0.01 \\ 0 & 0.80 & 0.90 & 0.99 \end{bmatrix}$$

$$\begin{array}{c} G \times (S \times R) \\ \uparrow grass \triangledown id \\ S \times R \\ \uparrow sprinkler \triangledown id \\ R \\ \uparrow rain \\ 1 \end{array}$$

The "same" state arrow

$$G \times (S \times R) \xleftarrow{state} 1 = (grass \triangledown id) \cdot (sprinkler \triangledown id) \cdot rain$$

but over a different **category** (next slide).

Bayesian network (probabilistic model)

$$G \times (S \times R) \xleftarrow{state} 1 \quad =$$

G	S	R	
dry	off	*no*	0.4800
		yes	0.0396
	on	*no*	0.0320
		yes	0.0000
wet	off	*no*	0.0000
		yes	0.1584
	on	*no*	0.2880
		yes	0.0020

Moreover, we can define

$$1 \xleftarrow{grass\_wet} G \times (S \times R) = [0\ 1] \cdot fst$$

$$1 \xleftarrow{raining} G \times (S \times R) = [0\ 1] \cdot snd \cdot snd$$

etc. to obtain e.g. $\mathrm{P}_{state}(grass\_wet) = grass\_wet \cdot state = 44.84\%$.

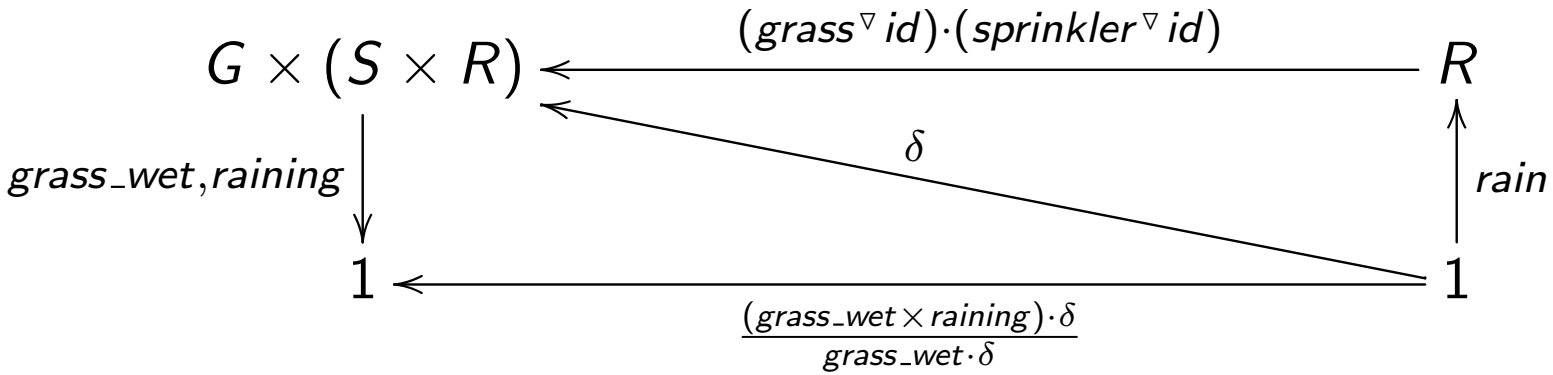
Bayesian network querying

Conditional probabilities over state distribution δ:

$$\mathrm{P}_\delta(a \mid b) = \frac{(a \times b) \cdot \delta}{b \cdot \delta} \quad \textit{where} \quad 1 \xleftarrow{a,b} S \xleftarrow{\delta} 1 \tag{4}$$

Boolean vectors a and b describe **event** sets.

Recall



Forwards: $\mathrm{P}_\delta(\textit{grass\_wet} \mid \textit{raining}) = 80.19\%$

Backwards: $\mathrm{P}_\delta(\textit{raining} \mid \textit{grass\_wet}) = 35.77\%$

By the way

Bayes theorem:

$$\mathrm{P}(a \mid b) = \mathrm{P}(b \mid a)\ \frac{\mathrm{P}(a)}{\mathrm{P}(b)} \tag{5}$$

cf. (assuming $\delta : 1 \to S$):

$$\mathrm{P}_\delta(a \mid b) = \mathrm{P}_\delta(b \mid a)\ \frac{\mathrm{P}_\delta(a)}{\mathrm{P}_\delta(b)}$$

$\Leftrightarrow$ { trivial }

$$\mathrm{P}_\delta(a \mid b)\ \mathrm{P}_\delta(b) = \mathrm{P}_\delta(b \mid a)\ \mathrm{P}_\delta(a)$$

$\Leftrightarrow$ { (4) twice }

$$(a \times b) \cdot \delta = (b \times a) \cdot \delta$$

$\square$

Towards probabilistic contracts

$\mathrm{P}_\delta(\mathit{raining} \mid \mathit{grass\_wet}) = 35.77\%$ — *backwards* reasoning — is suggestive of (weakest) **precondition** validation — in a sense, it tells how important raining is as **cause** for the grass to be wet (**effect**).

Note that probabilistic function

$$f : R \to S \times G$$
$$f = (\mathit{fst} \triangledown \mathit{grass}) \cdot (\mathit{sprinkler} \triangledown \mathit{id})$$

that is,

$f =$

		no	*yes*
off	*dry*	0.6	0.198
	wet	0	0.792
on	*dry*	0.04	0.010
	wet	0.36	0.001

describes a system that **reacts** to raining.

Towards probabilistic contracts

In what sense — measure? — can we say that some f satisfies the contract

$$grass\_wet \xleftarrow{f} raining \tag{6}$$

and what does (6) mean?

Back to **pure** function $f : Y \leftarrow X$:

$$q \xleftarrow{f} p \;\Leftrightarrow\; \langle \forall\ x\ :\ x \in X :\ p\ x \Rightarrow q\ (f\ x) \rangle$$

or, if you wish,

$$q \xleftarrow{f} p \;\Leftrightarrow\; \neg\ \langle \exists\ x\ :\ x \in X :\ p\ x \wedge \neg\ q\ (f\ x) \rangle$$

Towards probabilistic contracts

That is, model checking $q \xleftarrow{f} p$ means finding those $x \in X$ that violate the contract.

In a probabilistic setting, such $x \in X$ are captured by a **distribution** vector $\delta : 1 \to X$.

Term $q\ (f\ x)$ will then correspond to scalar $1 \xleftarrow{q \cdot f \cdot \delta} 1$ — a **probability**.

But first we have to regard f as a (kind of) **probabilistic relation**, as in the Bayesian network above.

That is, we need to have access to the I/O behaviour of f.

Towards probabilistic contracts

Recall that a **probabilistic** function $f : A \to B$ (PF) is half way between **pure** functions and **relations**: $! \cdot f = !$ holds and their support $\lfloor f \rfloor$ is a relation of the same type $A \to B$.

Below we will take PF

$f =$

	a_1	a_2	a_3
b_1	0.7	0.01	1
b_2	0.3	0.99	0

as example, with support

$\lfloor f \rfloor =$

	a_1	a_2	a_3
b_1	1	1	1
b_2	1	1	0

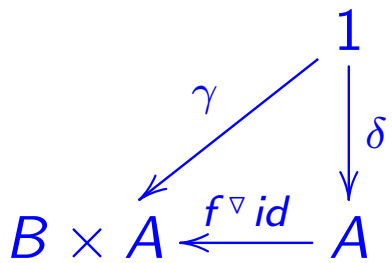
This support (a relation) can be mapped back to the PF

	a_1	a_2	a_3
b_1	0.5	0.5	1
b_2	0.5	0.5	0

as seen before.

Towards probabilistic contracts

The I/O behaviour of f knowing the distribution δ of the inputs is given by γ in



Example (f as before):

$\delta =$

A	
a_1	0.1
a_2	0.2
a_3	0.7

Then

$\gamma =$

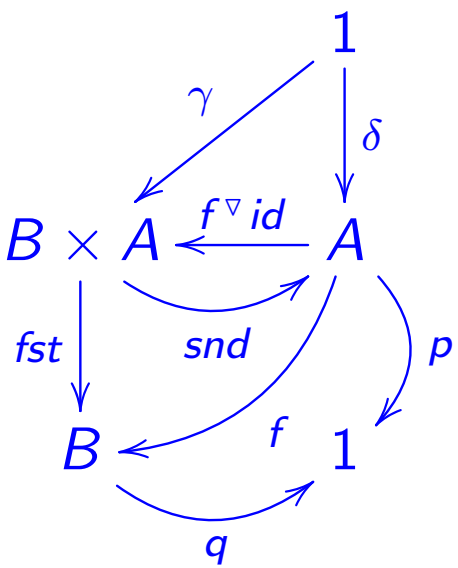
$B \times A$	
(b_1, a_1)	0.070
(b_1, a_2)	0.002
(b_1, a_3)	0.700
(b_2, a_1)	0.030
(b_2, a_2)	0.198
(b_2, a_3)	0

Measuring probabilistic contracts

Let us define:

$$[\![\ q \xleftarrow{f} p\ ]\!]_\delta = \mathrm{P}_\gamma(q \cdot \mathit{fst} \mid p \cdot \mathit{snd}) \tag{7}$$

where $\gamma = (f \triangledown \mathit{id}) \cdot \delta$ — check the diagram below:



In the next slide we show that (7) simplifies to

$$[\![\ q \xleftarrow{f} p\ ]\!]_\delta = (q \cdot f \times p) \cdot \frac{\delta}{p \cdot \delta} \tag{8}$$

where $\times$ is the Hadamard product.

Measuring probabilistic contracts

$$
\begin{aligned}
& [\![\ q \xleftarrow{f} p\ ]\!]_\delta \\
= & \quad \{\ \text{definition (7)}\ \} \\
& \mathrm{P}_{(f \triangledown id)\cdot\delta}(q \cdot fst \mid p \cdot snd) \\
= & \quad \{\ \text{definitions (4) and (13); } snd \cdot (f \triangledown id) = id\ \} \\
& \frac{(q \bowtie p) \cdot (f \triangledown id) \cdot \delta}{p \cdot \delta} \\
= & \quad \{\ (12)\ \} \\
& (q \cdot f \times p) \cdot \frac{\delta}{p \cdot \delta}
\end{aligned}
$$

Case $p = true$ simplifies to $[\![\ q \xleftarrow{f} true\ ]\!]_\delta = q \cdot f \cdot \delta$.

Measuring probabilistic contracts

Going pointwise:

$$[\![\ q \xleftarrow{f} p\ ]\!]_\delta = \frac{\langle \sum\ b, a\ :\ q\ b \wedge p\ a :\ (b\ f\ a)\ (\delta\ a)\rangle}{\langle \sum\ a\ :\ p\ a :\ \delta\ a\rangle}$$

Useful LA properties:

$$(M \triangledown N)^\circ \cdot (P \triangledown Q) = (M^\circ \cdot P) \times (N^\circ \cdot Q) \tag{9}$$

$$(P \bowtie Q)^\circ = P^\circ \triangledown Q^\circ \tag{10}$$

$$(M \bowtie N) \cdot (P \otimes Q) = M \cdot P \bowtie N \cdot Q \tag{11}$$

$$(P \bowtie Q) \cdot (F \triangledown G) = (P \cdot F) \times (Q \cdot G) \tag{12}$$

$$P \bowtie Q = (P \cdot fst) \times (Q \cdot snd) \tag{13}$$

Measuring probabilistic contracts

Example, recalling

$$f = \begin{array}{c|ccc} & a_1 & a_2 & a_3 \\ \hline b_1 & 0.7 & 0.01 & 1 \\ b_2 & 0.3 & 0.99 & 0 \end{array} \quad \text{and input} \quad \delta = \begin{array}{c|c} & \\ \hline a_1 & 0.1 \\ a_2 & 0.2 \\ a_3 & 0.7 \end{array}$$

Then, for instance,

$$\{b_2\} \xleftarrow{f} \{a_1, a_2\} = 76\%$$

$$\{b_2\} \xleftarrow{f} \{a_3\} = 0\%$$

$$\{b_2\} \xleftarrow{f} \mathit{true} = 22.8\%$$

$$\mathit{true} \xleftarrow{f} \{a_1, a_2\} = 100\%$$

etc

Model checking probabilistic contracts

In Alloy, given

```
assert contract { all a:A | p[a] => q[a.f] }
```

by executing

```
check contract for ... A
```

the tool will try and find a such that $p\ a \wedge \neg\ q\ (f\ a)$ holds.

Now let f be **probabilistic**. In the future one may imagine submitting something like

```
check contract >= 80% for ... A
```

hoping the tool cannot find δ such that $[\![\ q \xleftarrow{f} p\ ]\!]_\delta < 0.8$.

Towards a Probabilistic Alloy

Under the finite scope assumption, this boils down to solving classical systems of inequations:

$$[\![\ p \xleftarrow{f} q\ ]\!]_\delta < k$$

$$\Leftrightarrow \quad \{\ \text{recall } q \bowtie p = (q \cdot fst) \times (p \cdot snd)\ (13)\ \}$$

$$\frac{(q \bowtie p) \cdot (f \triangledown id) \cdot \delta}{p \cdot \delta} < k$$

$$\Leftrightarrow \quad \{\ \text{as before, re-arranging}\ \}$$

$$\langle \sum b, a \ :\ q\ b \wedge p\ a :\ (b\ f\ a)\ (\delta\ a) \rangle < \langle \sum a \ :\ p\ a :\ k\ \delta\ a \rangle$$

Example: find δ such that $[\![\ \{b_2\} \xleftarrow{f} \{a_1, a_2\}\ ]\!]_\delta < 0.8$ (next slide):

Towards a Probabilistic Alloy

$$\underbrace{b_2\ f\ a_1}_{0.3}\ \underbrace{\delta\ a_1}_{x} + \underbrace{b_2\ f\ a_2}_{0.99}\ \underbrace{\delta\ a_2}_{y} < 0.8\ \underbrace{\delta\ a_1}_{x} + 0.8\ \underbrace{\delta\ a_2}_{y}$$

$\Leftrightarrow$ { trivia }

$$0.19\ y < 0.5\ x$$

$\Leftrightarrow$ { $x + y = 1$ choosing $\delta\ a_3 = 0$ }

$$0.19\ (1 - x) < 0.5\ x$$

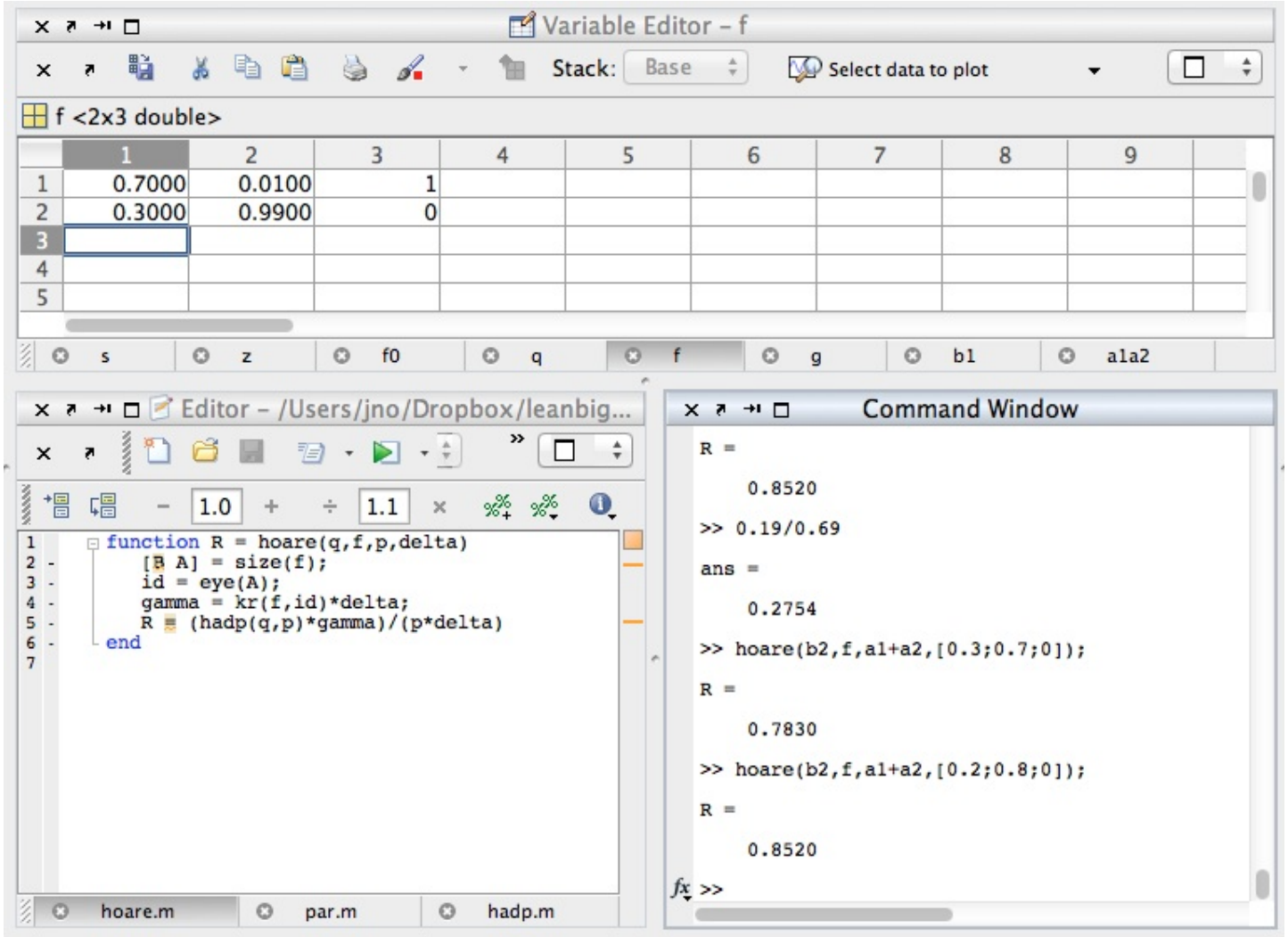
$\Leftrightarrow$ { solving }

$$\frac{0.19}{0.69} < x$$

□

Choose e.g. $\delta = \begin{bmatrix} 0.3 \\ 0.7 \\ 0 \end{bmatrix}$ and get $[\![\ \{b_2\} \xleftarrow{f} \{a_1, a_2\}\ ]\!]_\delta = 0.783$.

How I have worked thus far...



"Oldie but goodie"

Calculating probabilistic contracts

I didn't make a full sanity check of contract validity yet, but some expected **laws** are easy to check, cf. e.g.

$$[\![\ true \xleftarrow{f} p\ ]\!]_\delta$$

$$= \quad \{\ \text{unfold definition, } true = \top \text{ (1s everywhere)}\ \}$$

$$\frac{(\top \times p \cdot snd) \cdot (f \triangledown id) \cdot \delta}{p \cdot \delta}$$

$$= \quad \{\ \text{Hadamard product: } \top \times M = M,\ snd \cdot (f \triangledown id) = id\ \}$$

$$\frac{p \cdot \delta}{p \cdot \delta}$$

$$= \quad \{\ \text{trivia}\ \}$$

$$1$$

□

Calculating probabilistic contracts

But note that

$$[\![\ p \xleftarrow{g} \mathit{false}\ ]\!]_\delta = \frac{(p \cdot \mathit{fst} \times \bot \cdot \mathit{snd}) \cdot (g \triangledown \mathit{id}) \cdot \delta}{\bot \cdot \delta} = \frac{0}{0}$$

is mathematically undetermined...

Now let us have a look at the **sequence** $q \xleftarrow{f \cdot g} r$ of two contracts $q \xleftarrow{f} p$ and $p \xleftarrow{g} r$.

To begin with, let us handle the special case $q \xleftarrow{f} \mathit{true}$ and $p \xleftarrow{g} \mathit{true}$. We calculate (next slide):

Calculating probabilistic contracts

$$[\![\ p \xleftarrow{g} \mathit{true}\ ]\!]_\delta \leqslant 1$$

$\Rightarrow$ { multliply both sides by the same validity }

$$(q \cdot f \cdot g \cdot \delta) \times [\![\ p \xleftarrow{g} \mathit{true}\ ]\!]_\delta \leqslant q \cdot f \cdot g \cdot \delta$$

$\Leftrightarrow$ { associativity }

$$(q \cdot f \cdot (g \cdot \delta)) \times [\![\ p \xleftarrow{g} \mathit{true}\ ]\!]_\delta \leqslant q \cdot (f \cdot g) \cdot \delta$$

$\Leftrightarrow$ { definition (twice) }

$$[\![\ q \xleftarrow{f} \mathit{true}\ ]\!]_{g \cdot \delta} \times [\![\ p \xleftarrow{g} \mathit{true}\ ]\!]_\delta \leqslant [\![\ q \xleftarrow{f \cdot g} \mathit{true}\ ]\!]_\delta$$

Calculating probabilistic contracts

This suggests the generic inference rule:

$$[\![\, q \xleftarrow{f} p \,]\!]_{g\cdot\delta} \times [\![\, p \xleftarrow{g} r \,]\!]_{\delta} \leqslant [\![\, q \xleftarrow{f\cdot g} r \,]\!]_{\delta}$$

Example: f is as before and

$$g = \begin{array}{c|ccc} & c_1 & c_2 & c_3 \\ \hline a_1 & 0.4 & 0.2 & 0 \\ a_2 & 0 & 0.8 & 0 \\ a_3 & 0.6 & 0 & 1 \end{array} \quad \textit{for input} \quad \delta = \begin{array}{c|c} & \\ \hline c_1 & 0.1 \\ c_2 & 0.2 \\ c_3 & 0.7 \end{array}$$

We have $\{a_1, a_2\} \xleftarrow{g} \{c_2, c_3\} = 22\%$ for such δ.

Calculating probabilistic contracts

Because f receives outputs from g, the distribution of its inputs will be

$$g \cdot \delta = \begin{array}{c|c} & \\ \hline a_1 & 0.08 \\ a_2 & 0.16 \\ a_3 & 0.76 \end{array}$$

We measure $\{b_2\} \xleftarrow{f} \{a_1, a_2\} = 76\%$ for $g \cdot \delta$, the same by coincidence.

So the joint probability of both contracts holding is

$$\begin{aligned} & [\![\{b_2\} \xleftarrow{f} \{a_1, a_2\}]\!]_{g \cdot \delta} \times [\![\{a_1, a_2\} \xleftarrow{g} \{c_2, c_3\}]\!]_{\delta} \\ = \quad & 0.22 \times 0.76 \\ = \quad & 16.72\% \end{aligned}$$

This is smaller than the probability of the composite contract holding: $[\![\{b_2\} \xleftarrow{f \cdot g} \{c_2, c_3\}]\!]_{\delta} = 18.93\%$.

Calculating probabilistic contracts

However, this rule does not always hold (!), as the following counter-example shows:

$$\underbrace{[\![\ \{b_2\} \xleftarrow{f\cdot g} \{c_1\}\ ]\!]_\delta}_{12\%} < \underbrace{[\![\ \{b_2\} \xleftarrow{f} \{a_1, a_2\}\ ]\!]_{g\cdot\delta}}_{76\%} \times \underbrace{[\![\ \{a_1, a_2\} \xleftarrow{g} \{c_1\}\ ]\!]_\delta}_{40\%}$$

This recalls a similar problem identified long ago by McIver and Morgan (2005): *probabilistic* **Hoare triples** *not compositional* in general...

But our setting here is simpler (e.g. no demonic choice).

Current work

Currently checking this and other rules (calculating side conditions).

Further to (McIver and Morgan, 2005), there is recent work in the literature about **conditioning** in probabilistic programming, see e.g. (Gretz et al., 2015), which can be of help.

Also algorithms that use ideas from program analysis in probabilistic programming, see e.g. (Nori et al., 2014).

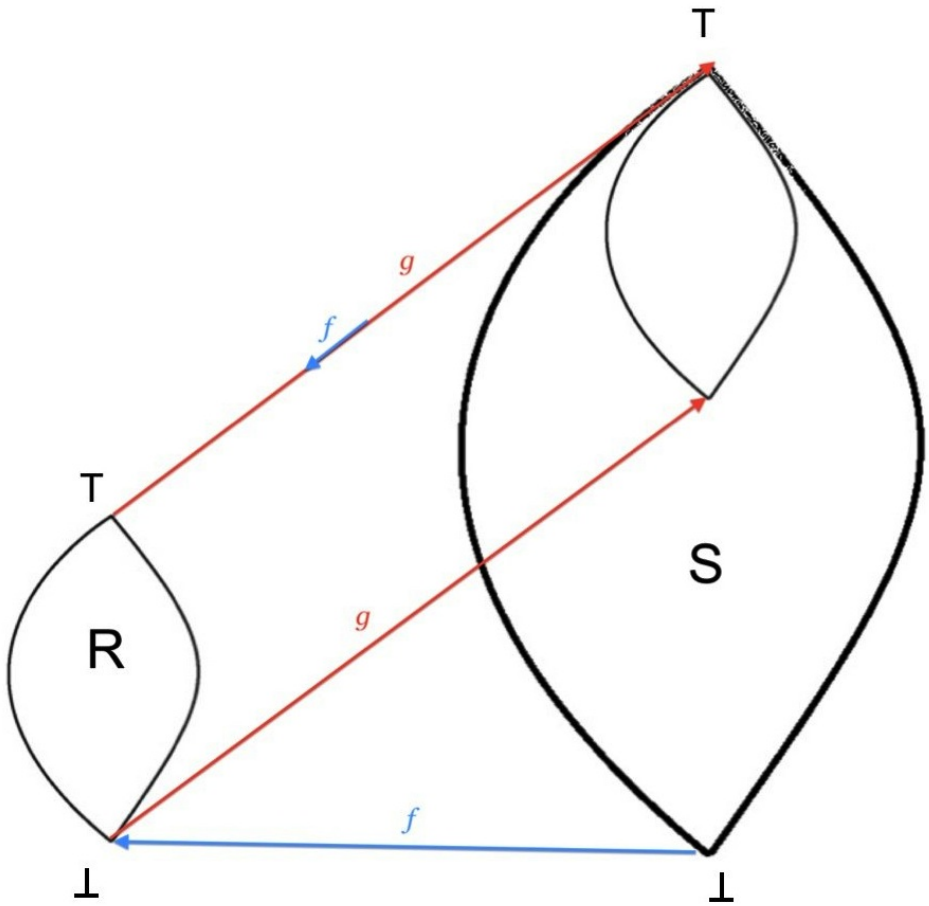
Carroll's suggestion at the meeting: McIver et al. (2008)

Afterthought

Recall matrix **supports**.

Can the current Alloy relational engine help in finding the counter-example **distributions**, as a kind of AI?

$f = \lfloor \_ \rfloor$, $g = \lceil \_ \rceil$ etc.



References

F. Gretz, N. Jansen, B.L. Kaminski, J.-P. Katoen, A. McIver, and F. Olmedo. Conditioning in probabilistic programming. *CoRR*, abs/1504.00198, 2015. URL `http://arxiv.org/abs/1504.00198`.

A. McIver and C. Morgan. *Abstraction, Refinement and Proof for Probabilistic Systems*. Monographs in Computer Science. Springer-Verlag, 2005. ISBN 0387401156.

A. K. McIver, C. C. Morgan, and C. Gonzalia. *Proofs and Refutations for Probabilistic Refinement*, pages 100–115. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. URL `http://dx.doi.org/10.1007/978-3-540-68237-0_9`.

A.V. Nori, C.-K. Hur, S.K. Rajamani, and S. Samuel. R2: an efficient MCMC sampler for probabilistic programs. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada.*, pages 2476–2482, 2014. URL `http://www.aaai.org/ocs/index.php/AAAI/AAAI14/paper/view/8192`.