Proving Uniformity and Independence by Self-Composition and Coupling

Gilles Barthe
Thomas Espitau
Benjamin Grégoire
Justin Hsu*
Pierre-Yves Strub

A puzzle

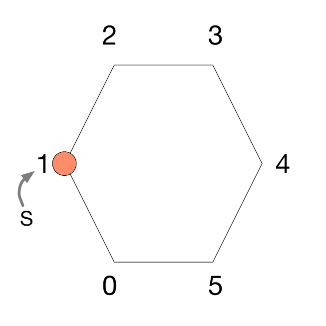
A random walk on a cycle

- Start at position $s \in \{0, 1, \ldots, n-1\}$
- Each iteration, flip a fair coin
  - Heads: increment position (modulo n)
  - Tails decrement position (modulo n)
- Return: last edge $(r, r+1)$ to be traversed

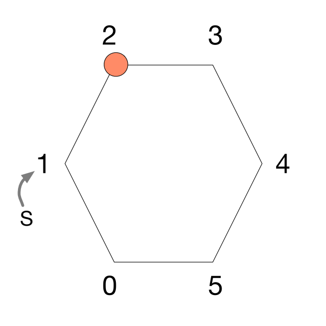
A question

What is the distribution of the returned edge, and how does it depend on the starting position s?

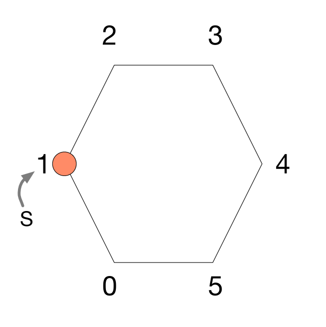
A puzzle



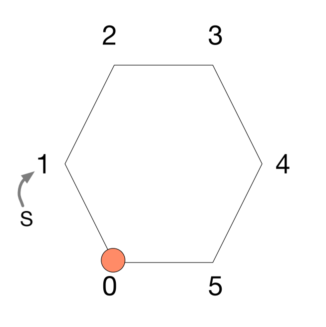
A puzzle



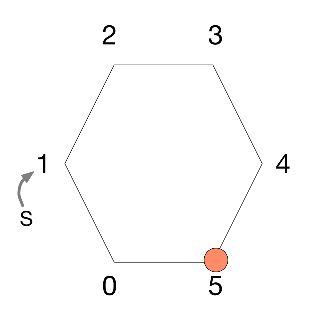
A puzzle



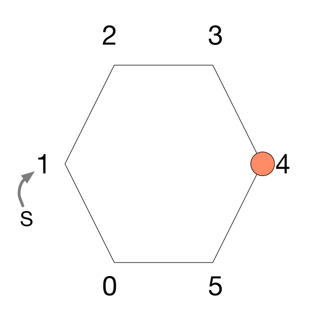
A puzzle



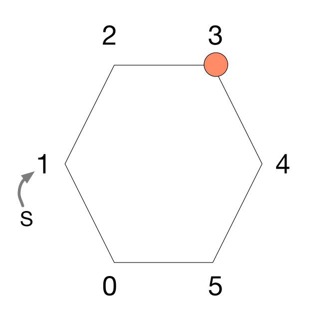
A puzzle



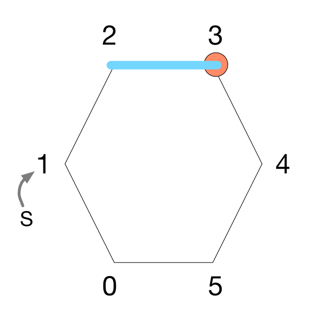
A puzzle



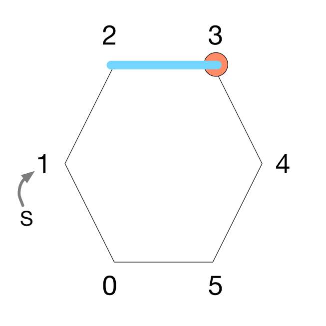
A puzzle



A puzzle



A puzzle



Somewhat surprisingly

Distribution of final edge is uniform:
Starting position s doesn't matter!

Basic properties of probabilistic programs

Uniformity of a variable X

For any two values w, v in the (finite) range of X, we have:

$$\Pr[X = w] = \Pr[X = v]$$

in output distribution.

Basic properties of probabilistic programs

Uniformity of a variable X

For any two values w, v in the (finite) range of X, we have:

$$\Pr[X = w] = \Pr[X = v]$$

in output distribution.

Independence of two variables X, Y

For any two values w, v, we have:

$$\Pr[X = w \wedge Y = v] = \Pr[X = w] \cdot \Pr[Y = v]$$

in output distribution.

Basic properties of probabilistic programs

Uniformity of a variable X

For any two values w, v in the (finite) range of X, we have:

$$\Pr[X = w] = \Pr[X = v]$$

in output distribution.

Independence of two variables X, Y

For any two values w, v, we have:

$$\Pr[X = w \wedge Y = v] = \Pr[X = w] \cdot \Pr[Y = v]$$

in output distribution.

Can be quite subtle to verify!

The idea today

Use logic for relational verification
to verify uniformity
and independence

A crash course: the relational logic pRHL

A curious program logic: pRHL [Barthe, Grégoire, Zanella-Béguelin]

pWhile: An imperative language with random sampling

$$c ::= x \leftarrow e \mid x \xleftarrow{\$} \mathit{flip}(p) \mid \textsf{if } e \textsf{ then } c \textsf{ else } c \mid \textsf{while } e \textsf{ do } c \mid \textsf{skip} \mid c;\ c$$

A curious program logic: pRHL [Barthe, Grégoire, Zanella-Béguelin]

pWhile: An imperative language with random sampling

$c ::= x \leftarrow e \mid \boxed{x \xleftarrow{\$} \mathit{flip}(p)} \mid \text{if } e \text{ then } c \text{ else } c \mid \text{while } e \text{ do } c \mid \text{skip} \mid c;\ c$

pRHL is a program logic that is:

- Probabilistic: Programs can draw samples

A curious program logic: pRHL [Barthe, Grégoire, Zanella-Béguelin]

pWhile: An imperative language with random sampling

$$c ::= x \leftarrow e \mid x \xleftarrow{\$} \mathit{flip}(p) \mid \text{if } e \text{ then } c \text{ else } c \mid \text{while } e \text{ do } c \mid \text{skip} \mid c;\ c$$

pRHL is a program logic that is:

- Probabilistic: Programs can draw samples
- Relational: Describe executions of two programs

Judgments in pRHL

$$\boxed{\{P(in\langle 1\rangle, in\langle 2\rangle)\}\ c \sim c'\ \{Q(out\langle 1\rangle, out\langle 2\rangle)\}}$$

Judgments in pRHL

$$\boxed{\{P(in\langle 1\rangle, in\langle 2\rangle)\}\ c \sim c'\ \{Q(out\langle 1\rangle, out\langle 2\rangle)\}}$$

Assertions

- Non-probabilistic
- FO formulas over program variables tagged with $\langle 1\rangle$ or $\langle 2\rangle$

Judgments in pRHL

$$\boxed{\{P(in\langle 1\rangle, in\langle 2\rangle)\}\ c \sim c'\ \{Q(out\langle 1\rangle, out\langle 2\rangle)\}}$$

Assertions

- Non-probabilistic
- FO formulas over program variables tagged with $\langle 1\rangle$ or $\langle 2\rangle$

Judgments in pRHL

$$\{P(in\langle 1\rangle, in\langle 2\rangle)\}\ c \sim c'\ \{Q(out\langle 1\rangle, out\langle 2\rangle)\}$$

Assertions

- Non-probabilistic
- FO formulas over program variables tagged with $\langle 1\rangle$ or $\langle 2\rangle$

Deep connection to probabilistic couplings

- Proofs specify how to correlate random samplings in runs
- Reduce sources of randomness, simplify verification

For our purposes today: equality of distributions

If this is provable:

$$\vdash \{P\}\ c \sim c'\ \{e\langle 1\rangle = e'\langle 2\rangle\}$$

Then:

On any two input memories related by P, the distribution of e in the first output is equal to the distribution of e' in the second output.

In particular: express equality of probabilities

If this is provable for booleans b, b':

$$\vdash \{P\}\ c \sim c'\ \{b\langle 1\rangle = b'\langle 2\rangle\}$$

Then:

On any two input memories related by P, the probability of b in the first output is equal to the probability of b' in the second output.

Random sampling rules in pRHL

Simplified version

$$\textsc{FlipEq}\ \frac{}{\vdash \{\top\}\ \ x \xleftarrow{\$} \mathit{flip}(p) \sim x' \xleftarrow{\$} \mathit{flip}(p)\ \ \{x\langle 1\rangle = x'\langle 2\rangle\}}$$

$$\textsc{FlipNeg}\ \frac{}{\vdash \{\top\}\ \ x \xleftarrow{\$} \mathit{flip}(p) \sim x' \xleftarrow{\$} \mathit{flip}(1-p)\ \ \{x\langle 1\rangle = \neg x'\langle 2\rangle\}}$$

Random sampling rules in pRHL

Simplified version

$$\textsc{FlipEq}\ \frac{}{\vdash \{\top\}\ \ x \xleftarrow{\$} \mathit{flip}(p) \sim x' \xleftarrow{\$} \mathit{flip}(p)\ \ \{x\langle 1\rangle = x'\langle 2\rangle\}}$$

$$\textsc{FlipNeg}\ \frac{}{\vdash \{\top\}\ \ x \xleftarrow{\$} \mathit{flip}(p) \sim x' \xleftarrow{\$} \mathit{flip}(1-p)\ \ \{x\langle 1\rangle = \neg x'\langle 2\rangle\}}$$

Reading: for any $p \in [0, 1]$,

1. [FlipEq]: Distributions of $\mathit{flip}(p)$ and $\mathit{flip}(p)$ are equal
2. [FlipNeg]: Distributions of $\mathit{flip}(p)$ and negated $\mathit{flip}(1-p)$ are equal

Rest of rules are standard (≈ Hoare logic)

Assignments

$$\textsc{Assn}\ \frac{}{\vdash \{Q[e\langle 1\rangle, e'\langle 2\rangle / x\langle 1\rangle, x'\langle 2\rangle]\}\ \ x \leftarrow e_1 \sim x' \leftarrow e_2\ \ \{Q\}}$$

Sequencing

$$\textsc{Seq}\ \frac{\vdash \{P\}\ \ c_1 \sim c_1'\ \ \{Q\} \qquad \vdash \{Q\}\ \ c_2 \sim c_2'\ \ \{R\}}{\vdash \{P\}\ \ c_1;\, c_2 \sim c_1';\, c_2'\ \ \{R\}}$$

Loops

$$\textsc{While}\ \frac{\vdash \{P \wedge b\langle 1\rangle\}\ \ c \sim c'\ \ \{P\} \qquad \models P \implies b\langle 1\rangle = b'\langle 2\rangle}{\vdash \{P\}\ \ \text{while } b \text{ do } c \sim \text{while } b' \text{ do } c'\ \ \{P \wedge \neg b\langle 1\rangle\}}$$

Rest of rules are standard (≈ Hoare logic)

Assignments

$$\textsc{Assn}\ \frac{}{\vdash \{Q[e\langle 1\rangle, e'\langle 2\rangle / x\langle 1\rangle, x'\langle 2\rangle]\}\ \ x \leftarrow e_1 \sim x' \leftarrow e_2\ \ \{Q\}}$$

Sequencing

$$\textsc{Seq}\ \frac{\vdash \{P\}\ \ c_1 \sim c_1'\ \ \{Q\} \qquad \vdash \{Q\}\ \ c_2 \sim c_2'\ \ \{R\}}{\vdash \{P\}\ \ c_1; c_2 \sim c_1'; c_2'\ \ \{R\}}$$

Loops

$$\textsc{While}\ \frac{\vdash \{P \land b\langle 1\rangle\}\ \ c \sim c'\ \ \{P\} \qquad \models P \implies b\langle 1\rangle = b'\langle 2\rangle}{\vdash \{P\}\ \ \text{while } b \text{ do } c \sim \text{while } b' \text{ do } c'\ \ \{P \land \neg b\langle 1\rangle\}}$$

Benefits of pRHL

Probabilistic properties without probabilistic reasoning

- Abstract away all probabilities
- All reasoning is about relation between samples

Highly similar to Hoare logic

- Most things “just work”
- Compositional reasoning

Benefits of pRHL

Probabilistic properties without probabilistic reasoning

- Abstract away all probabilities
- All reasoning is about relation between samples

Highly similar to Hoare logic

- Most things “just work”
- Compositional reasoning

Apply to non-relational properties, like uniformity and independence.

Verifying uniformity: simulating a fair coin

The algorithm

Goal

Generate one fair coin flip, using only coin flips with a fixed bias $p \in (0, 1)$.

Procedure

1. Flip two coins with bias p
2. Re-flip as long as they are equal
3. Return the first coin flip the first time they are different

In code

Consider the program fair:

$$
\begin{aligned}
& x \leftarrow tt; \\
& y \leftarrow tt; \\
& \text{while } x = y \text{ do} \\
& \qquad x \xleftarrow{\$} \mathit{flip}(p); \\
& \qquad y \xleftarrow{\$} \mathit{flip}(p); \\
& \text{return}(x)
\end{aligned}
$$

To show: generates fair coin flip

Distribution of return value is uniform

Observation: uniformity can be proved in pRHL

For every two booleans w, v, show:

$$\vdash \{p\langle 1\rangle = p\langle 2\rangle\}\ \ \mathit{fair} \sim \mathit{fair}\ \ \{(x\langle 1\rangle = w) \iff (x\langle 2\rangle = v)\}$$

Reading: for every two booleans w, v,

$\Pr[x = w] = \Pr[x = v]$ in the output of fair.

Four choices in all for w, v

- We show the cases with $w \neq v$

Step 1: rearrange program

Two equivalent programs: fair and fair'

$$
\begin{array}{l}
x \leftarrow tt; \\
y \leftarrow tt; \\
\text{while } x = y \text{ do} \\
\quad x \xleftarrow{\$} \mathit{flip}(p); \\
\quad y \xleftarrow{\$} \mathit{flip}(p); \\
\text{return}(x)
\end{array}
\qquad
\begin{array}{l}
x \leftarrow tt; \\
y \leftarrow tt; \\
\text{while } x = y \text{ do} \\
\quad y \xleftarrow{\$} \mathit{flip}(p); \\
\quad x \xleftarrow{\$} \mathit{flip}(p); \\
\text{return}(x)
\end{array}
$$

Step 1: rearrange program

Two equivalent programs: fair and fair'

$$
\begin{array}{ll}
x \leftarrow tt; \\
y \leftarrow tt; \\
\textsf{while } x = y \textsf{ do} \\
\quad x \xleftarrow{\$} \mathit{flip}(p); \\
\quad y \xleftarrow{\$} \mathit{flip}(p); \\
\textsf{return}(x)
\end{array}
\qquad\qquad
\begin{array}{ll}
x \leftarrow tt; \\
y \leftarrow tt; \\
\textsf{while } x = y \textsf{ do} \\
\quad y \xleftarrow{\$} \mathit{flip}(p); \\
\quad x \xleftarrow{\$} \mathit{flip}(p); \\
\textsf{return}(x)
\end{array}
$$

Step 1: rearrange program

Two equivalent programs: fair and fair'

$$
\begin{array}{l}
x \leftarrow tt; \\
y \leftarrow tt; \\
\text{while } x = y \text{ do} \\
\quad x \xleftarrow{\$} \mathit{flip}(p); \\
\quad y \xleftarrow{\$} \mathit{flip}(p); \\
\text{return}(x)
\end{array}
\qquad
\begin{array}{l}
x \leftarrow tt; \\
y \leftarrow tt; \\
\text{while } x = y \text{ do} \\
\quad y \xleftarrow{\$} \mathit{flip}(p); \\
\quad x \xleftarrow{\$} \mathit{flip}(p); \\
\text{return}(x)
\end{array}
$$

Step 1: rearrange program

Two equivalent programs: fair and fair'

$$\begin{array}{ll}
x \leftarrow tt; & x \leftarrow tt; \\
y \leftarrow tt; & y \leftarrow tt; \\
\textsf{while } x = y \textsf{ do} & \textsf{while } x = y \textsf{ do} \\
\quad x \xleftarrow{\$} \mathit{flip}(p); & \quad y \xleftarrow{\$} \mathit{flip}(p); \\
\quad y \xleftarrow{\$} \mathit{flip}(p); & \quad x \xleftarrow{\$} \mathit{flip}(p); \\
\textsf{return}(x) & \textsf{return}(x)
\end{array}$$

For the cases $w \neq v$, suffices to show:

$$\boxed{\vdash \{p\langle 1\rangle = p\langle 2\rangle\} \ \ \mathit{fair} \sim \mathit{fair}' \ \ \{x\langle 1\rangle = \neg x\langle 2\rangle\}}$$

Step 2: apply the loop rule

$$
\begin{array}{l}
\text{while } x = y \text{ do} \\
\qquad x \xleftarrow{\$} \mathit{flip}(p); \\
\qquad y \xleftarrow{\$} \mathit{flip}(p); \\
\text{return}(x)
\end{array}
\qquad\qquad
\begin{array}{l}
\text{while } x = y \text{ do} \\
\qquad y \xleftarrow{\$} \mathit{flip}(p); \\
\qquad x \xleftarrow{\$} \mathit{flip}(p); \\
\text{return}(x)
\end{array}
$$

Step 2: apply the loop rule

$$\begin{array}{l} \text{while } x = y \text{ do} \\ \quad x \xleftarrow{\$} \mathit{flip}(p); \\ \quad y \xleftarrow{\$} \mathit{flip}(p); \\ \text{return}(x) \end{array} \qquad \begin{array}{l} \text{while } x = y \text{ do} \\ \quad y \xleftarrow{\$} \mathit{flip}(p); \\ \quad x \xleftarrow{\$} \mathit{flip}(p); \\ \text{return}(x) \end{array}$$

In the body: apply [FLIPEQ] for both pairs of samples

Step 2: apply the loop rule

$$\begin{array}{l}\text{while } x = y \text{ do}\\ \quad x \xleftarrow{\$} \mathit{flip}(p);\\ \quad y \xleftarrow{\$} \mathit{flip}(p);\\ \text{return}(x)\end{array} \qquad \begin{array}{l}\text{while } x = y \text{ do}\\ \quad y \xleftarrow{\$} \mathit{flip}(p);\\ \quad x \xleftarrow{\$} \mathit{flip}(p);\\ \text{return}(x)\end{array}$$

In the body: apply [FlipEq] for both pairs of samples

- We have: $x\langle 1\rangle = y\langle 2\rangle$

Step 2: apply the loop rule

while $x = y$ do
  $x \xleftarrow{\$} \mathit{flip}(p)$;
  $y \xleftarrow{\$} \mathit{flip}(p)$;
return(x)

while $x = y$ do
  $y \xleftarrow{\$} \mathit{flip}(p)$;
  $x \xleftarrow{\$} \mathit{flip}(p)$;
return(x)

In the body: apply [FlipEq] for both pairs of samples

- We have: $x\langle 1\rangle = y\langle 2\rangle$
- And: $x\langle 2\rangle = y\langle 1\rangle$

Step 2: apply the loop rule

$$\begin{array}{l}\text{while } x = y \text{ do}\\ \quad x \xleftarrow{\$} \mathit{flip}(p);\\ \quad y \xleftarrow{\$} \mathit{flip}(p);\\ \text{return}(x)\end{array} \qquad \begin{array}{l}\text{while } x = y \text{ do}\\ \quad y \xleftarrow{\$} \mathit{flip}(p);\\ \quad x \xleftarrow{\$} \mathit{flip}(p);\\ \text{return}(x)\end{array}$$

In the body: apply [FLIPEQ] for both pairs of samples

- We have: $x\langle 1\rangle = y\langle 2\rangle$
- And: $x\langle 2\rangle = y\langle 1\rangle$

Establishes main invariant:

$$\boxed{x\langle 2\rangle = (\text{if } x\langle 1\rangle = y\langle 1\rangle \text{ then } y\langle 2\rangle \text{ else } \neg x\langle 1\rangle)}$$

Step 3: putting it all together

Applying [Assn], [Seq] shows:

$$\boxed{\vdash \{p\langle 1\rangle = p\langle 2\rangle\}\ \ \mathit{fair} \sim \mathit{fair}\ \ \{(x\langle 1\rangle = w) \iff (x\langle 2\rangle = v)\}}$$

when $w \neq v$; can also show same judgment when $w = v$.

Conclude

$$\boxed{\mathit{fair} \text{ returns a uniform boolean}}$$

Extensions: verifying independence

Verifying independence: the easier way

Observation: reduce independence to uniformity

$$\boxed{\begin{gathered}(x, y) \text{ is uniform over pairs} \\ \Downarrow \\ x \text{ and } y \text{ are independent}\end{gathered}}$$

Limitation

- Only can show independence for uniform variables

Verifying independence: the harder way

Use self-composition

- Let $c[1], c[2]$ be two copies of c with disjoint variables
- Prove a pRHL judgment relating

$$c \sim c[1]; c[2]$$

Verifying independence: the harder way

Use self-composition

- Let $c[1], c[2]$ be two copies of c with disjoint variables
- Prove a pRHL judgment relating

$$c \sim c[1]; c[2]$$

Independence of two variables X, Y

For any two values w, v, we have:

$$\Pr[X = w \wedge Y = v] = \Pr[X = w] \cdot \Pr[Y = v]$$

in output distribution.

Verifying independence: the harder way

Use self-composition

- Let $c[1], c[2]$ be two copies of c with disjoint variables
- Prove a pRHL judgment relating

$$c \sim c[1]; c[2]$$

Independence of two variables X, Y

For any two values w, v, we have:

$$\Pr[X = w \wedge Y = v] = \Pr[X = w] \cdot \Pr[Y = v]$$

in output distribution.

Verifying independence: the harder way

Use self-composition

- Let $c[1], c[2]$ be two copies of c with disjoint variables
- Prove a pRHL judgment relating

$$c \sim c[1]; c[2]$$

Independence of two variables X, Y

For any two values w, v, we have:

$$\Pr[X = w \wedge Y = v] = \Pr[X = w] \cdot \Pr[Y = v]$$

in output distribution.

Verifying independence: the harder way

Use self-composition

- Let $c[1], c[2]$ be two copies of c with disjoint variables
- Prove a pRHL judgment relating

$$c \sim c[1]; c[2]$$

Independence of two variables X, Y

For any two values w, v, we have:

$$\Pr[X = w \wedge Y = v] = \Pr[X = w] \cdot \Pr[Y = v]$$

in output distribution.

Verifying independence: the harder way

Use self-composition

- Let $c[1], c[2]$ be two copies of c with disjoint variables
- Prove a pRHL judgment relating

$$c \sim c[1]; c[2]$$

Independence of two variables X, Y

For any two values w, v, we have:

$$\Pr[X = w \wedge Y = v] = \Pr[X = w] \cdot \Pr[Y = v]$$

in output distribution.

Benefits

- Can prove independence for non-uniform variables
- Similar ideas can cover conditional independence

Summing up

See the paper for

Lots more examples

- Cycle random walk
- Pairwise and k-wise independence
- Bayesian network
- Ballot theorem

Details about the implementation

- Most examples formalized in EasyCrypt framework

Future directions

- Automate this approach
- Explore relational verification for non-relational properties
- Integrate with more general probabilistic verification tools

Proving Uniformity and Independence by Self-Composition and Coupling

Gilles Barthe
Thomas Espitau
Benjamin Grégoire
Justin Hsu*
Pierre-Yves Strub