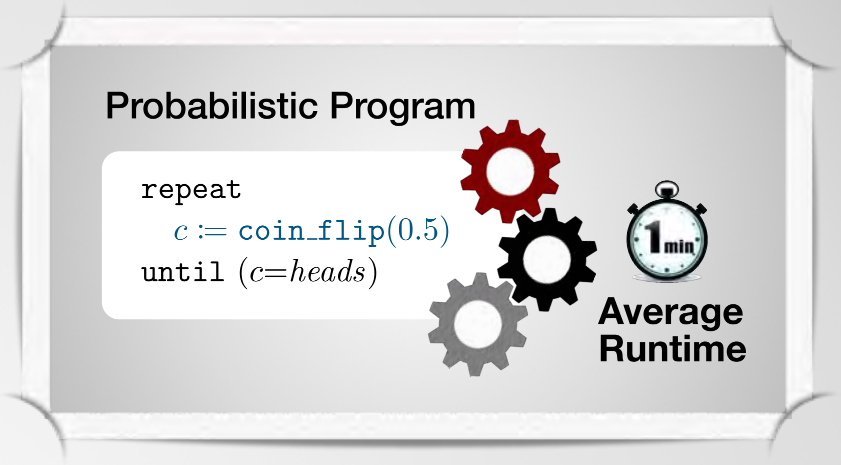
Run-Time Analysis of Probabilistic Programs



Probabilistic Program

```
repeat
  c := coin_flip(0.5)
until (c=heads)
```

Average Runtime

Joost-Pieter Katoen

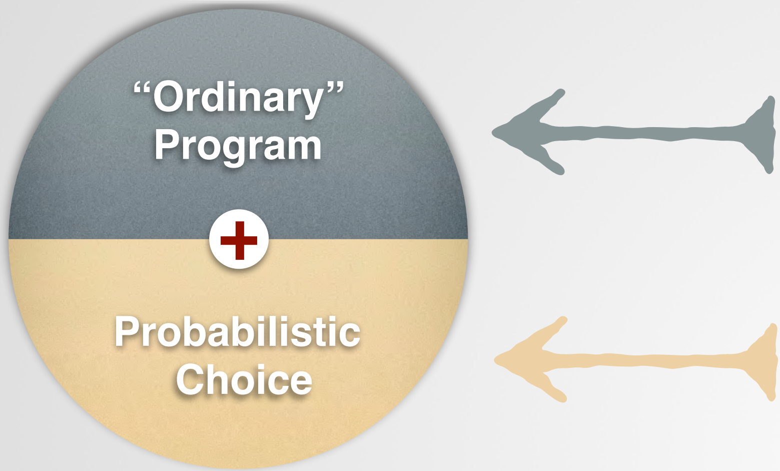
joint work with: Benjamin Kaminski, Christoph Matheja, and Federico Olmedo

Lehrstuhl für Informatik 2 Softwaremodellierung und Verifikation | RWTH AACHEN UNIVERSITY

IFIP WG2.2 Meeting Singapore, September 2016

Probabilistic Programs — Basics

What is a probabilistic program?



“Ordinary” Program

+

Probabilistic Choice

- **imperative** In this work
- functional
- logical
- …

- randomly choose a process with which communicate
- select a random prime in interval *$[1, n^2]$*
- flip a (fair/biased) coin;

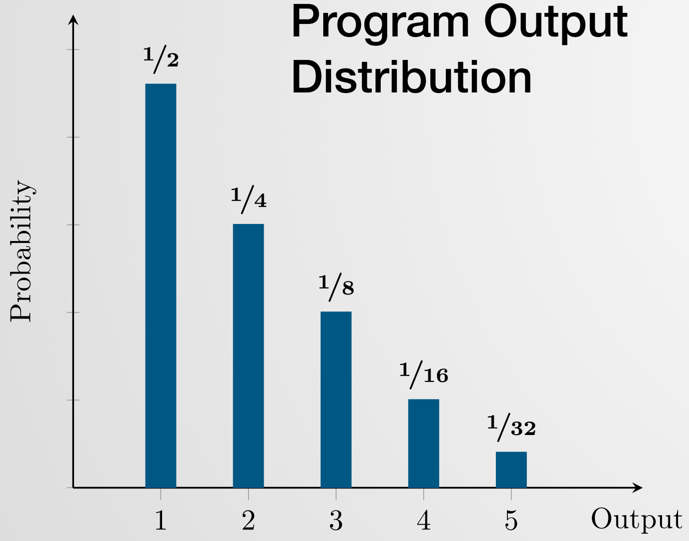
Program behaviour (**input-output relation** + runtime) is determined by the outcome of its probabilistic choices.

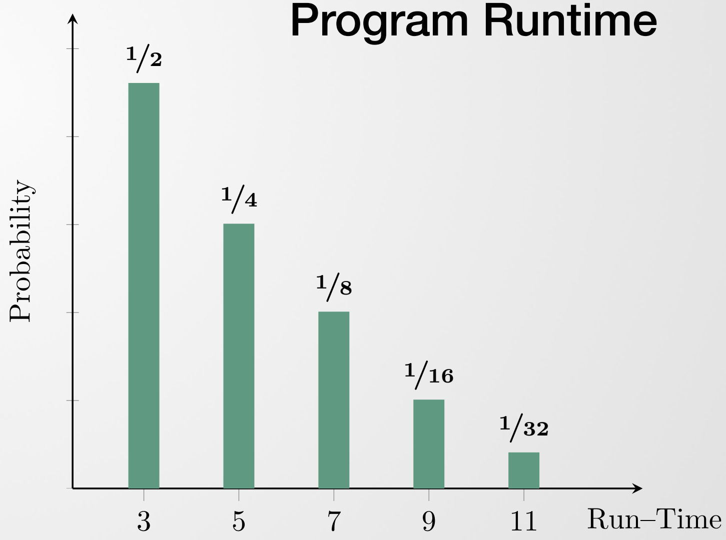
- Program output is a probability distribution: v_1 with probability p_1, v_2 with probability p_2, etc
- Program runtime is a random variable: t_1 with probability p_1, t_2 with probability p_2, etc

Probabilistic Programs — Example

Probabilistic program that simulates a geometric distribution

```
$C_{\text{geo}}$:  $n := 0$;
         repeat
           $n := n + 1$;
           $c :=$ coin_flip$(0.5)$
         until $(c{=}heads)$;
         return $n$
```


Program Output Distribution


Program Runtime

Average Runtime:

$$3 \cdot \frac{1}{2} + 5 \cdot \frac{1}{4} + \cdots + (2n{+}1) \cdot \frac{1}{2^n} + \cdots = 5$$

Randomisation Allows Speeding Up Algorithms

Quicksort:

```
$\mathsf{QS}(A) \triangleq$
    if $(|A| \leq 1)$ then return $(A)$;
    $i := \lfloor |A|/2 \rfloor$;
    $A_< := \{a' \in A \mid a' < A[i]\}$;
    $A_> := \{a' \in A \mid a' > A[i]\}$;
    return $\big(\mathsf{QS}(A_<) +\!\!+ A[i] +\!\!+ \mathsf{QS}(A_>)\big)$
```

Worst case complexity:

$O(n^2)$ comparisons

Randomised Quicksort:

```
$\mathsf{rQS}(A) \triangleq$
    if $(|A| \leq 1)$ then return $(A)$;
    $i := \mathsf{rand}[1 \ldots |A|]$;
    $A_< := \{a' \in A \mid a' < A[i]\}$;
    $A_> := \{a' \in A \mid a' > A[i]\}$;
    return $\big(\mathsf{QS}(A_<) +\!\!+ A[i] +\!\!+ \mathsf{QS}(A_>)\big)$
```

Worst case complexity:

$O(n \log(n))$ expected comparisons

Run-Time Analysis of Probabilistic Programs is Intricate

- Probabilistic programs may admit infinite runs, but finite expected run-time

```
$C_{\text{geo}}$:  $n := 0$;
       repeat
          $n := n+1$; $c :=$ coin_flip$(0.5)$
       until $(c = \mathit{heads})$
```

- Positive almost sure termination is not closed under sequential composition:

```
$C_1$:  $x := 1$;
     repeat
        $c :=$ coin_flip$(0.5)$; $x := 2x$;
     until $(c = \mathit{heads})$
```

```
$C_2$:  repeat
          $x := x-1$;
        until $(x \leq 0)$
```

C_1 and C_2 both terminate in finite expected time, while C_1;C_2 does not.

- (Positive) almost-sure termination is “more undecidable” than ordinary termination

This talk

- wp-Calculus for bounding the expected runtime of probabilistic programs
- Soundness of the calculus w.r.t. an operational program semantics
- Consistency w.r.t. Nielson's logic for bounding runtime of ordinary programs
- Runtime analysis of random walk and the coupon collector’s problem

Probabilistic Programming Language

C	$::=$	$\texttt{empty}$	empty program
	$\mid$	$\texttt{skip}$	effectless operation
	$\mid$	$\texttt{halt}$	immediate termination
	$\mid$	$x :\approx \boldsymbol{\mu}$	probabilistic assignment
	$\mid$	$C;\ C$	sequential composition
	$\mid$	$\{C\} \,\square\, \{C\}$	non–deterministic choice
	$\mid$	$\texttt{if}\,(\boldsymbol{\eta})\,\{C\}\,\texttt{else}\,\{C\}$	probabilistic conditional
	$\mid$	$\texttt{while}\,(\boldsymbol{\eta})\,\{C\}$	probabilistic while loop

- Truncated geometric distribution:

```
if $(\frac{1}{2}\cdot\langle\mathsf{true}\rangle + \frac{1}{2}\cdot\langle\mathsf{false}\rangle)$ $\{succ :\approx \mathsf{true}\}$ else
    $\{$if $(\frac{1}{2}\cdot\langle\mathsf{true}\rangle + \frac{1}{2}\cdot\langle\mathsf{false}\rangle)$ $\{succ :\approx \mathsf{true}\}$
        else $\{succ :\approx \mathsf{false}\}\}$
```

- Race between tortoise and hare:

```
$h :\approx 0;\ t :\approx 30;$
while $(h \leq t)$
  $t :\approx t + 1;$
  if $(\frac{1}{2}\cdot\langle\mathsf{true}\rangle + \frac{1}{2}\cdot\langle\mathsf{false}\rangle)$ $\{h :\approx h + \texttt{Unif}[0..10]\}$
      else $\{$empty$\}$
```

The Expected Runtime Transformer — Basics

Our aim:

program *C* → $h_C : \overbrace{\mathcal{S} \to \mathbb{R}^{\infty}_{\geq 0}}^{\mathbb{T}}$

$h_C(s) \mapsto$ number of $\mathtt{skip}$s, assignments and guard evaluations in the execution of *C* from state *s*

Our approach:

We use a continuation passing style through transformer

$$\mathsf{ert}[C] : \mathbb{T} \to \mathbb{T}$$

$f \mapsto$ runtime of the computation following program *C* → $\mathsf{ert}\,[C]\,(f) \mapsto$ runtime of *C*, plus the computation following *C*

In particular,

$\mathsf{ert}\,[C]\,(\mathbf{0})\,(s) \;\mapsto$ runtime of *C*, when started in state *s.*

The Expected Runtime Transformer — Inductive Definition

$$
\begin{aligned}
\mathsf{ert}\,[\texttt{empty}]\,(f) &= f \\
\mathsf{ert}\,[\texttt{skip}]\,(f) &= \mathbf{1} + f \\
\mathsf{ert}\,[\texttt{halt}]\,(f) &= \mathbf{0}
\end{aligned}
$$

$$\sum_v \mathsf{Pr}[\mu{=}v] \cdot f(s[x/v])$$

$$
\begin{aligned}
\mathsf{ert}\,[x \;:\approx\; \textstyle\sum_i p_i \cdot \langle v_i \rangle]\,(f) &= \mathbf{1} + \lambda s \bullet\; \textstyle\sum_i p_i \cdot f(s[x \mapsto v_i]) \\
\mathsf{ert}\,[C_1;\; C_2]\,(f) &= \mathsf{ert}\,[C_1]\,(\mathsf{ert}\,[C_2]\,(f)) \\
\mathsf{ert}\,[\{C_1\} \,\square\, \{C_2\}]\,(f) &= \max\{\mathsf{ert}\,[C_1]\,(f)\,,\, \mathsf{ert}\,[C_2]\,(f)\} \\
\mathsf{ert}\,[\texttt{if}\,(\eta)\,\{C_1\}\,\texttt{else}\,\{C_2\}]\,(f) &= \mathbf{1} + \mathsf{Pr}\,[\eta{=}\mathsf{true}] \cdot \mathsf{ert}\,[C_1]\,(f) + \mathsf{Pr}\,[\eta{=}\mathsf{false}] \cdot \mathsf{ert}\,[C_2]\,(f) \\
\mathsf{ert}\,[\texttt{while}\,(\eta)\,\{C\}]\,(f) &= \mathit{lfp}\,\left(F_f^{\langle \eta, C \rangle}\right) \quad \text{where} \\
&\; F_f^{\langle \eta, C \rangle}(X) = \mathbf{1} + \mathsf{Pr}\,[\eta{=}\mathsf{false}] \cdot f + \mathsf{Pr}\,[\eta{=}\mathsf{true}] \cdot \mathsf{ert}\,[C]\,(X)
\end{aligned}
$$

Characteristic functional

The Expected Runtime Transformer — Elementary Properties

Monotonicity:

$$f \preceq g \quad \Longrightarrow \quad \mathsf{ert}\,[C]\,(f) \;\preceq\; \mathsf{ert}\,[C]\,(g)$$

Propagation of constants:

$$\mathsf{ert}\,[C]\,(\mathbf{k} + f) \;=\; \mathbf{k} + \mathsf{ert}\,[C]\,(f)$$

provided C is $\mathtt{halt}$–free

Preservation of ∞:

$$\mathsf{ert}\,[C]\,(\infty) \;=\; \infty$$

provided C is $\mathtt{halt}$–free

Sub-additivity:

$$\mathsf{ert}\,[C]\,(f + g) \;\preceq\; \mathsf{ert}\,[C]\,(f) + \mathsf{ert}\,[C]\,(g);$$

C is fully probabilistic

Scaling:

$$\mathsf{ert}\,[C]\,(r \cdot f) \;\succeq\; \min\{1,\, r\} \cdot \mathsf{ert}\,[C]\,(f)$$

$$\mathsf{ert}\,[C]\,(r \cdot f) \;\preceq\; \max\{1,\, r\} \cdot \mathsf{ert}\,[C]\,(f)$$

The Expected Runtime Transformer — Application Example

$$\begin{aligned}&\mathsf{ert}\,[x :\approx \textstyle\sum_i p_i \cdot \langle v_i \rangle]\,(f) \;=\\ &\qquad \mathbf{1} + \lambda s \bullet \textstyle\sum_i p_i \cdot f(s[x \mapsto v_i])\\ &\mathsf{ert}\,[\texttt{if}\;(\eta)\,\{C_1\}\,\texttt{else}\,\{C_2\}]\,(f) \;=\\ &\qquad \mathbf{1} + \mathsf{Pr}\,[\eta{=}\mathsf{true}] \cdot \mathsf{ert}\,[C_1]\,(f) + \mathsf{Pr}\,[\eta{=}\mathsf{false}] \cdot \mathsf{ert}\,[C_2]\,(f)\end{aligned}$$

$$\begin{aligned}C_{\mathsf{trunc}}\mathbf{:}\quad &\texttt{if}\;\left(\tfrac{1}{2}\cdot\langle\mathsf{true}\rangle + \tfrac{1}{2}\cdot\langle\mathsf{false}\rangle\right)\;\{\mathit{succ} :\approx \mathsf{true}\}\;\texttt{else}\\ &\quad \Big\{\texttt{if}\;\left(\tfrac{1}{2}\cdot\langle\mathsf{true}\rangle + \tfrac{1}{2}\cdot\langle\mathsf{false}\rangle\right)\;\{\mathit{succ} :\approx \mathsf{true}\}\\ &\qquad\quad \texttt{else}\;\{\mathit{succ} :\approx \mathsf{false}\}\Big\}\end{aligned}$$

$$\begin{aligned}\mathsf{ert}\,[C_{\mathsf{trunc}}]\,(\mathbf{0}) \;&=\; \mathbf{1} + \tfrac{1}{2}\cdot\mathsf{ert}\,[\mathit{succ} :\approx \mathsf{true}]\,(\mathbf{0})\\ &\qquad + \tfrac{1}{2}\cdot\mathsf{ert}\,[\texttt{if}\;(\ldots)\;\{\mathit{succ} :\approx \mathsf{true}\}\;\texttt{else}\;\{\mathit{succ} :\approx \mathsf{false}\}]\,(\mathbf{0})\\ &=\; \mathbf{1} + \tfrac{1}{2}\cdot\mathbf{1} + \tfrac{1}{2}\cdot\Big(\mathbf{1}+\tfrac{1}{2}\cdot\mathsf{ert}\,[\mathit{succ} :\approx \mathsf{true}]\,(\mathbf{0}) + \tfrac{1}{2}\cdot\mathsf{ert}\,[\mathit{succ} :\approx \mathsf{false}]\,(\mathbf{0})\Big)\\ &=\; \mathbf{1} + \tfrac{1}{2}\cdot\mathbf{1} + \tfrac{1}{2}\cdot\Big(\mathbf{1} + \tfrac{1}{2}\cdot\mathbf{1} + \tfrac{1}{2}\cdot\mathbf{1}\Big) \;=\; \frac{\mathbf{5}}{\mathbf{2}}\end{aligned}$$

$\therefore$ The execution of C_{trunc} takes, on average, 2.5 units of time

The Expected Runtime of Loops — Proof Rules

$$\frac{F_f^{\langle \eta, C\rangle}(I) \leq I}{\mathsf{ert}\left[\mathtt{while}\,(\eta)\,\{C\}\right](f) \leq I}\ [\mathsf{while}]$$

$$F_f^{\langle \eta, C\rangle}(X) \;=\; \mathbf{1} + \Pr\left[\eta{=}\mathsf{false}\right] \cdot f + \Pr\left[\eta{=}\mathsf{true}\right] \cdot \mathsf{ert}\left[C\right](X)$$

$$\frac{F_f^{\langle \eta, C\rangle}(\mathbf{0}) \geq I_0 \qquad F_f^{\langle \eta, C\rangle}(I_n) \geq I_{n+1}}{\mathsf{ert}\left[\mathtt{while}\,(\eta)\,\{C\}\right](f) \geq \lim_{n\to\infty} I_n}\ [\omega\text{–}\mathsf{while}^{\geq}]$$

$$\frac{F_f^{\langle \eta, C\rangle}(\mathbf{0}) \leq I_0 \qquad F_f^{\langle \eta, C\rangle}(I_n) \leq I_{n+1}}{\mathsf{ert}\left[\mathtt{while}\,(\eta)\,\{C\}\right](f) \leq \lim_{n\to\infty} I_n}\ [\omega\text{–}\mathsf{while}^{\leq}]$$

Theorem

The above proof rules are sound and complete.

The Expected Runtime of Loops — Example

$$\frac{\mathbf{1} + \mathsf{Pr}\,[\eta{=}\mathsf{false}] \cdot f + \mathsf{Pr}\,[\eta{=}\mathsf{true}] \cdot \mathsf{ert}\,[C]\,(I) \;\leq\; I}{\mathsf{ert}\,[\texttt{while}\,(\eta)\,\{C\}]\,(f) \leq I}\;[\mathsf{while}]$$

$$\mathsf{ert}\,[x :\approx \mu]\,(f) \;=\; \mathbf{1} + \lambda s \bullet\, \mathsf{E}_\mu\,(\lambda v.\, f(s[x/v]))$$

$$C_{\mathsf{geo}^\star}: \quad \texttt{while}\,(b = 1)\,\{b :\approx \tfrac{1}{2}\cdot\langle 0\rangle + \tfrac{1}{2}\cdot\langle 1\rangle\}$$

To upper-bound the runtime of $C_{\mathsf{geo}^\star}$ we apply rule [while] with continuation $f \;=\; \mathbf{0}$ and invariant $I \;=\; \mathbf{1} + [b = 1] \cdot \mathbf{4}$

$$\begin{aligned}
&\mathbf{1} + [b \neq 1] \cdot \mathbf{0} + [b = 1] \cdot \mathsf{ert}\left[b :\approx \tfrac{1}{2}\cdot\langle 0\rangle + \tfrac{1}{2}\cdot\langle 1\rangle\right](I) \\
&\quad = \;\mathbf{1} + [b = 1] \cdot \left(\mathbf{1} + \tfrac{1}{2} \cdot I[b/0] + \tfrac{1}{2} \cdot I[b/1]\right) \\
&\quad = \;\mathbf{1} + [b = 1] \cdot \Big(\mathbf{1} + \tfrac{1}{2} \cdot \underbrace{(\mathbf{1} + [0 = 1] \cdot \mathbf{4})}_{=\mathbf{1}} + \tfrac{1}{2} \cdot \underbrace{(\mathbf{1} + [1 = 1] \cdot \mathbf{4})}_{=\mathbf{5}}\Big) \\
&\quad = \;\mathbf{1} + [b = 1] \cdot \mathbf{4} \;=\; I \;\leq\; I
\end{aligned}$$

and conclude that $\mathsf{ert}\,[C_{\mathsf{geo}^\star}]\,(\mathbf{0}) \;\leq\; \mathbf{1} + [b{=}1] \cdot \mathbf{4}$

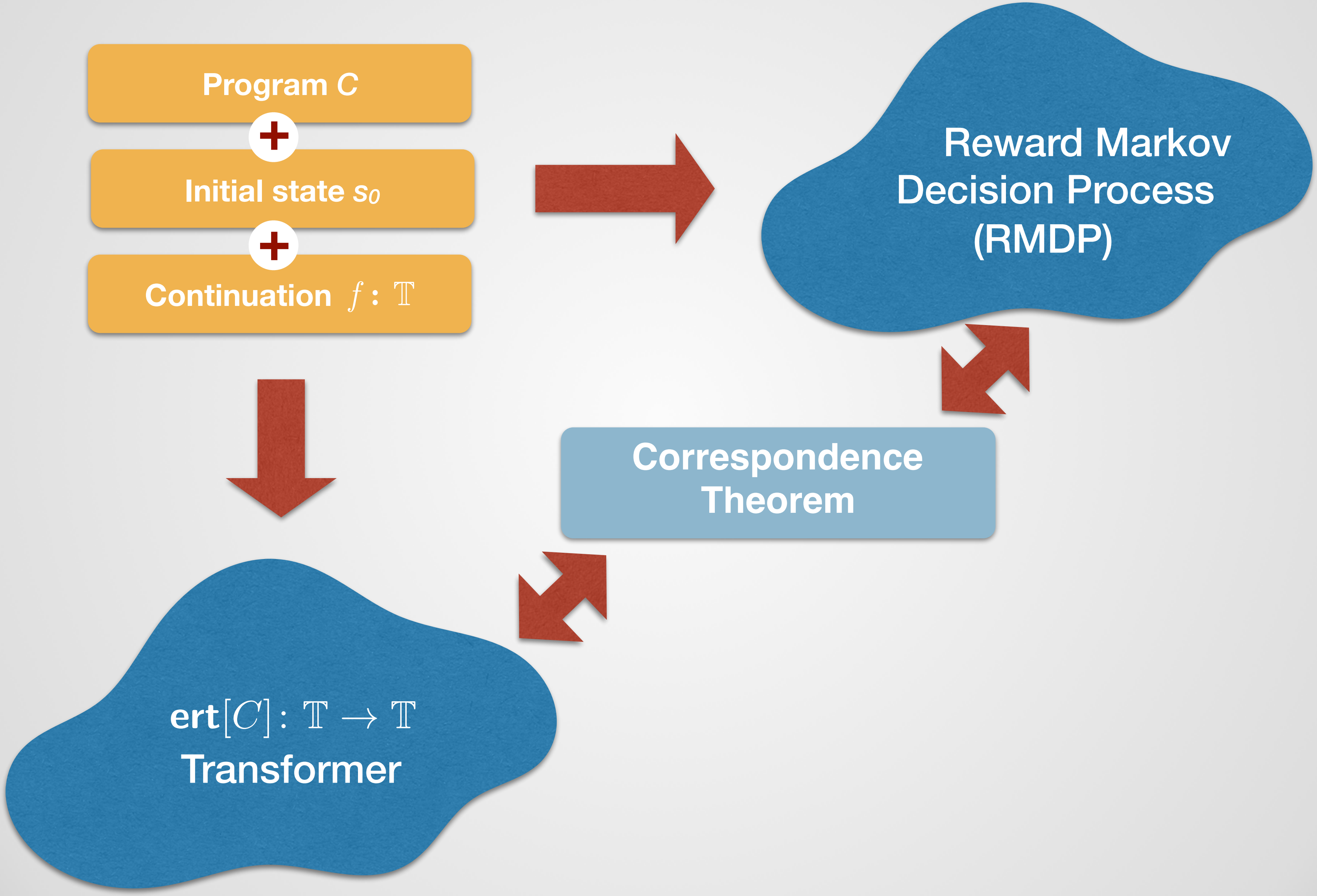
$\therefore$ The expected runtime of $C_{\mathsf{geo}^\star}$ is at most 5 from any initial state where b=1 and at most 1 from all other states.

The Expected Runtime of Loops — Bound Refinement

$$\frac{\mathsf{ert}\left[\,\texttt{while}\,(\eta)\,\{C\}\right](f) \leq g \qquad F_f^{\langle \eta, C\rangle}(g) \leq g}{\mathsf{ert}\left[\texttt{while}\,(\eta)\,\{C\}\right](f) \;\leq\; F_f^{\langle \eta, C\rangle}(g) \;\leq\; g}$$

$$\frac{\mathsf{ert}\left[\,\texttt{while}\,(\eta)\,\{C\}\right](f) \geq g \qquad F_f^{\langle \eta, C\rangle}(g) \geq g}{\mathsf{ert}\left[\texttt{while}\,(\eta)\,\{C\}\right](f) \;\geq\; F_f^{\langle \eta, C\rangle}(g) \;\geq\; g}$$

Operational Program Semantics — The Big Picture



Program C

Initial state s_0

Continuation $f : \mathbb{T}$

Reward Markov Decision Process (RMDP)

Correspondence Theorem

$\mathsf{ert}[C] : \mathbb{T} \to \mathbb{T}$

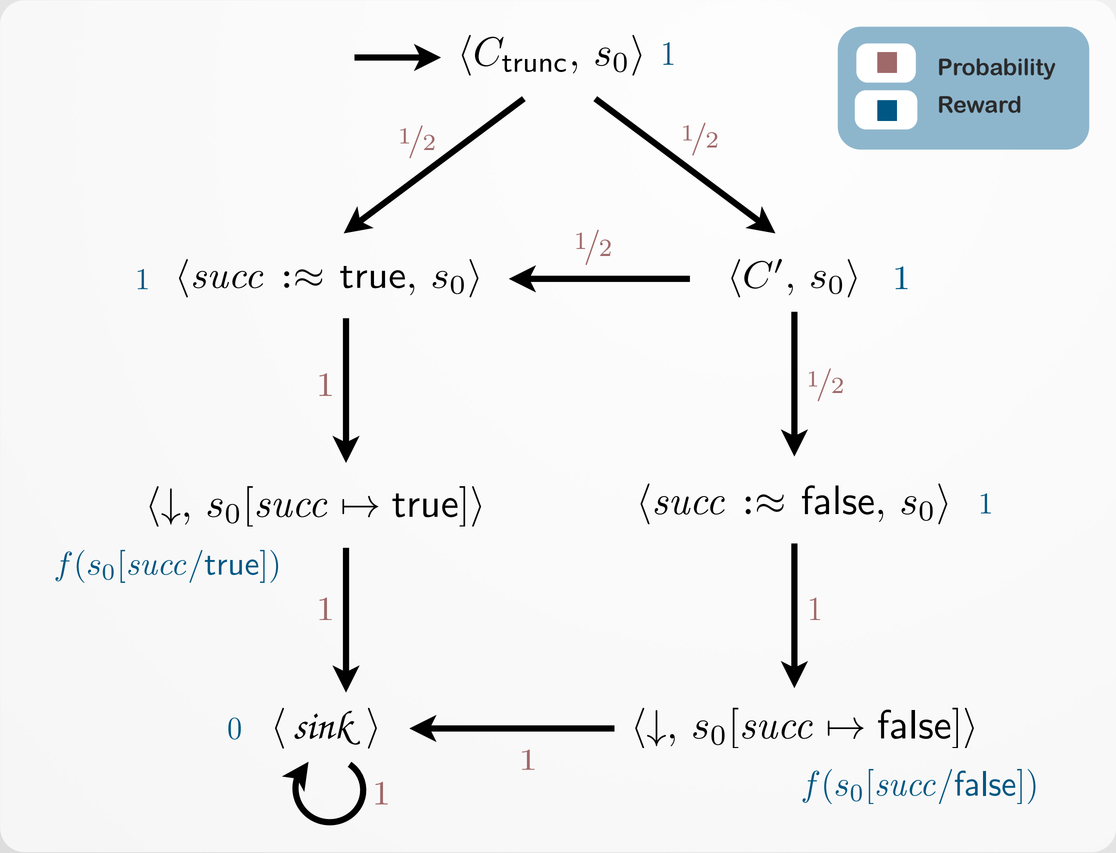
Transformer

Operational Program Semantics

$$C_{\textsf{trunc}}\textbf{:}\quad \texttt{if}\ \left(\tfrac{1}{2}\cdot\langle\textsf{true}\rangle + \tfrac{1}{2}\cdot\langle\textsf{false}\rangle\right)\ \{\mathit{succ} :\approx \textsf{true}\}\ \texttt{else}$$

$$\underbrace{\left\{\texttt{if}\ \left(\tfrac{1}{2}\cdot\langle\textsf{true}\rangle + \tfrac{1}{2}\cdot\langle\textsf{false}\rangle\right)\ \{\mathit{succ} :\approx \textsf{true}\}\ \texttt{else}\ \{\mathit{succ} :\approx \textsf{false}\}\right\}}_{C'}$$

RMDP for program $C_{\textsf{trunc}}$, initial state *s₀* and continuation *f*



🟥 Probability
🟦 Reward
$\langle C_{\textsf{trunc}},\ s_0\rangle$ 1
1/2
1/2
1 $\langle \mathit{succ} :\approx \textsf{true},\ s_0\rangle$
1/2
$\langle C',\ s_0\rangle$ 1
1
1/2
$\langle \downarrow,\ s_0[\mathit{succ} \mapsto \textsf{true}]\rangle$
$f(s_0[\mathit{succ}/\textsf{true}])$
$\langle \mathit{succ} :\approx \textsf{false},\ s_0\rangle$ 1
1
1
0 $\langle\ \mathit{sink}\ \rangle$
1
1
$\langle \downarrow,\ s_0[\mathit{succ} \mapsto \textsf{false}]\rangle$
$f(s_0[\mathit{succ}/\textsf{false}])$

Operational Program Semantics — Reward MDP Construction

RMDP State	Interpretation	RMDP Reward
$\langle C,\, s\rangle, \langle {\downarrow}; C,\, s\rangle$	intermediate execution point (*C* remaining computation from intermediate state *s)*	0 or 1
$\langle {\downarrow},\, s\rangle$	normal termination (s final program state)	*f(s)*
$\langle\, \mathit{sink}\, \rangle$	termination (normal or halt)	0

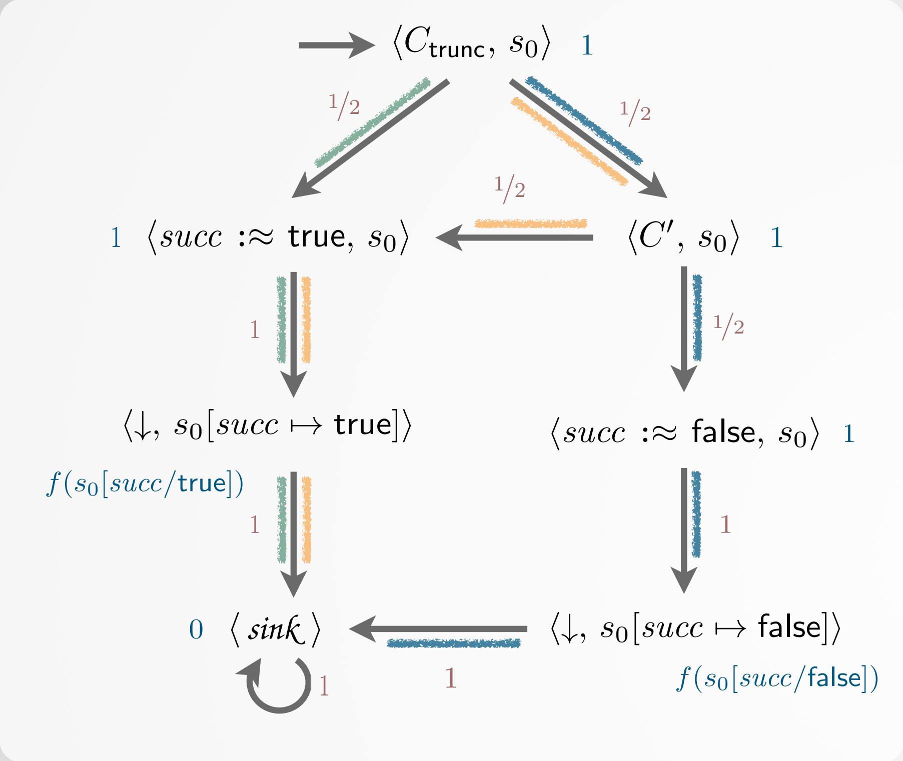
Sample Construction Rules

$$\frac{\mathsf{Pr}\left[\mu(s){=}v\right] = p > 0}{\langle x \ :\approx\ \mu,\, s\rangle \xrightarrow{\tau} \langle {\downarrow},\, s[x/v]\rangle \vdash p}\ [\text{pr–assgn}]$$

$$\frac{\mathsf{Pr}\left[\eta(s){=}\mathsf{true}\right] = p > 0}{\langle \mathtt{if}\,(\eta)\,\{C_1\}\,\mathtt{else}\,\{C_2\},\, s\rangle \xrightarrow{\tau} \langle C_1,\, s\rangle \vdash p}\ [\text{if–true}]$$

$$\frac{}{\langle \mathtt{while}\,(\eta)\,\{C\},\, s\rangle \xrightarrow{\tau} \langle \mathtt{if}\,(\eta)\,\{C; \mathtt{while}\,(\eta)\,\{C\}\}\,\mathtt{else}\,\{\mathtt{empty}\},\, s\rangle \vdash 1}\ [\text{while}]$$

Operational Program Semantics — Relation to the $\mathsf{ert}\left[\cdot\right]$ Transformer



$\langle C_{\mathsf{trunc}},\ s_0\rangle$ 1
1/2
1/2
1/2
1 $\langle succ :\approx \mathsf{true},\ s_0\rangle$
$\langle C',\ s_0\rangle$ 1
1
1/2
$\langle\downarrow,\ s_0[succ \mapsto \mathsf{true}]\rangle$
$\langle succ :\approx \mathsf{false},\ s_0\rangle$ 1
$f(s_0[succ/\mathsf{true}])$
1
1
0 $\langle\ sink\ \rangle$
$\langle\downarrow,\ s_0[succ \mapsto \mathsf{false}]\rangle$
1
1
$f(s_0[succ/\mathsf{false}])$

$$
\begin{aligned}
&\mathsf{ExpRew}^{\mathcal{M}^f_s[C]}(\langle\ sink\ \rangle)\\
&\quad= \mathsf{Pr}[\pi_{\mathsf{true}}]\cdot rew(\pi_{\mathsf{true}})\\
&\qquad+ \mathsf{Pr}[\pi_{\mathsf{false\ \ true}}]\cdot rew(\pi_{\mathsf{false\ \ true}})\\
&\qquad+ \mathsf{Pr}[\pi_{\mathsf{false\ \ false}}]\cdot rew(\pi_{\mathsf{false\ \ false}})\\
&\quad= \left(\tfrac{1}{2}\cdot 1\cdot 1\right)\cdot\left(1+1+f(s_0[succ/\mathsf{true}])\right)\\
&\qquad+ \left(\tfrac{1}{2}\cdot\tfrac{1}{2}\cdot 1\cdot 1\right)\cdot\left(1+1+1+f(s_0[succ/\mathsf{true}])\right)\\
&\qquad+ \left(\tfrac{1}{2}\cdot\tfrac{1}{2}\cdot 1\cdot 1\right)\cdot\left(1+1+1+f(s_0[succ/\mathsf{false}])\right)\\
&\quad= \tfrac{5}{2}\ +\ \tfrac{3}{4}\cdot f(s_0[succ/\mathsf{true}])\ +\ \tfrac{1}{4}\cdot f(s_0[succ/\mathsf{false}]).
\end{aligned}
$$

$$
\therefore\quad \mathsf{ExpRew}^{\mathcal{M}^{\mathbf{0}}_{s_0}[C]}(\langle\ sink\ \rangle)\ =\ \tfrac{5}{2}\ =\ \mathsf{ert}\left[C\right](\mathbf{0})\,(s_0)
$$

Theorem (Soundness)

Let $\mathsf{ExpRew}^{\mathcal{M}^f_{s_0}[C]}(\langle\ sink\ \rangle)$ be the expected reward to reach the sink in the RMDP associated to program *C*, initial state *s₀* and continuation *f*. Then

$$
\mathsf{ert}\left[C\right](f)\,(s_0)\ =\ \mathsf{ExpRew}^{\mathcal{M}^f_{s_0}[C]}(\langle\ sink\ \rangle)\ .
$$

Nielson's Logic for Deterministic Programs — Basics

Judgments

(Total Correctness) (Runtime Bound)

$$\{P\}\ \boxed{C}\ \{\boxed{E} \Downarrow Q\} \ \triangleq\ \{P\}\ C\ \{\Downarrow Q\}\ +\ \begin{array}{c} C \text{ terminates from } s \text{ in (at most a} \\ \text{mult. of) } [\![E]\!](s) \text{ steps if } s \models P \end{array}$$

deterministic program

numeric expression over program variables

Eg. $\{\mathtt{true}\}\ \ \mathtt{while}\,(x \geq 0)\,\{x := x-1\}\ \ \{x\ \Downarrow\ x<0\}$

Sample proof rules

$$\frac{}{\vdash_E \{Q[x/e]\}\ x := e\ \{1\ \Downarrow\ Q\}}\ [\text{Assgn}]$$

$$\frac{\vdash_E \{P \wedge B\}\ C_1\ \{E\ \Downarrow\ Q\} \qquad \vdash_E \{P \wedge \neg B\}\ C_2\ \{E\ \Downarrow\ Q\}}{\vdash_E \{P\}\ \mathtt{if}\,(B)\,\{C_1\}\,\mathtt{else}\,\{C_2\}\ \{E\ \Downarrow\ Q\}}\ [\text{if}]$$

Consistency w.r.t. Nielson's Logic

Theorem $\text{ert}[\cdot]$ generalises Nielson's logic to probabilistic programs

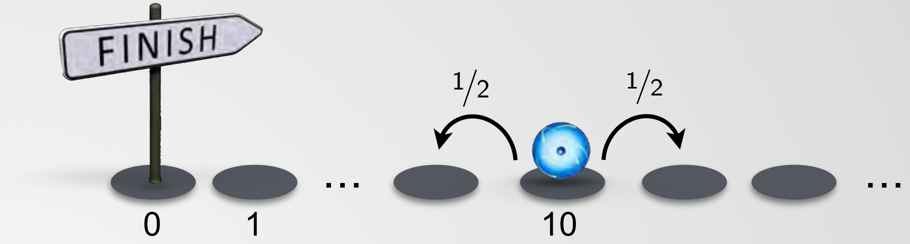
For any deterministic program C,

$$\vdash \{P\}\ C\ \{\Downarrow\ Q\} \quad \Longrightarrow \quad \vdash_E \{P\}\ C\ \{\text{ert}\,[C]\,(\mathbf{0})\ \Downarrow\ Q\} \qquad \text{(soundness)}$$

$$\vdash_E \{P\}\ C\ \{E\ \Downarrow\ Q\} \quad \Longrightarrow \quad \exists k \bullet\ \text{ert}\,[C]\,(\mathbf{0})\,(s) = k \cdot [\![E]\!](s) \qquad \text{(completeness)}$$

Runtime Analysis of a Random Walk

A particle starts at position *x*=10 and moves with equal probability to the left or to the right in each turn. The random walk stops when the particle reaches position *x*=0.



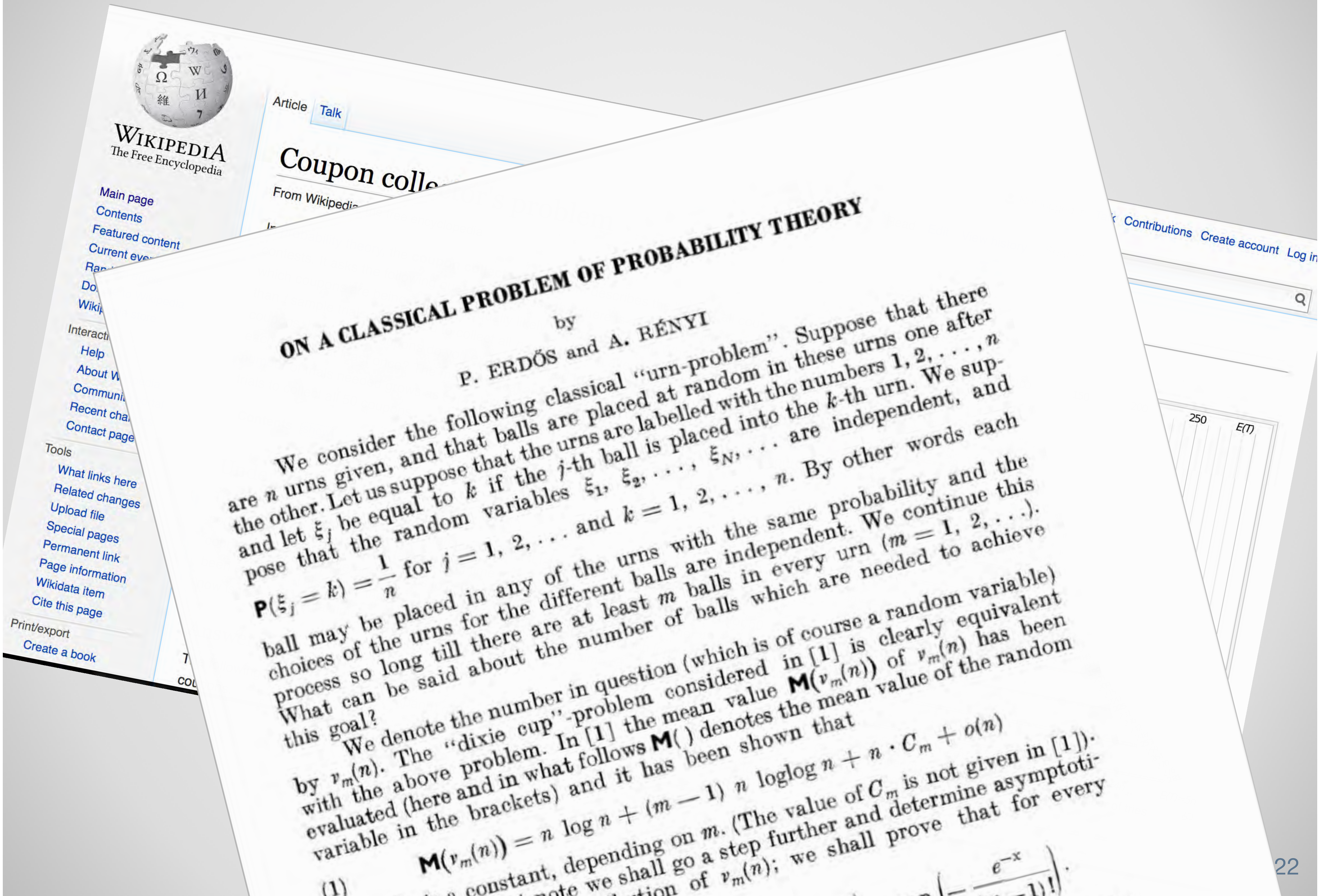
What is the expected number of moves to termination?

$$C_{\text{rw}}\colon \quad x \coloneqq 10;\ \texttt{while}\,(x > 0)\,\{x :\approx {}^1\!/\!_2 \cdot \langle x{-}1\rangle + {}^1\!/\!_2 \cdot \langle x{+}1\rangle\}$$

It can be shown that C_{rw} terminates with probability one. Using our expected runtime calculus one can show that it takes an expected infinite time to do so:

$$\mathsf{ert}\,[C_{\text{rw}}]\,(\mathbf{0}) \;=\; \infty$$

Coupon Collector’s Problem



ON A CLASSICAL PROBLEM OF PROBABILITY THEORY

by

P. ERDŐS and A. RÉNYI

We consider the following classical "urn-problem". Suppose that there are n urns given, and that balls are placed at random in these urns one after the other. Let us suppose that the urns are labelled with the numbers $1, 2, \ldots, n$ and let ξ_j be equal to k if the j-th ball is placed into the k-th urn. We suppose that the random variables $\xi_1, \xi_2, \ldots, \xi_N, \ldots$ are independent, and

$$\mathbf{P}(\xi_j = k) = \frac{1}{n} \text{ for } j = 1, 2, \ldots \text{ and } k = 1, 2, \ldots, n.$$

By other words each ball may be placed in any of the urns with the same probability and the choices of the urns for the different balls are independent. We continue this process so long till there are at least m balls in every urn ($m = 1, 2, \ldots$). What can be said about the number of balls which are needed to achieve this goal?

We denote the number in question (which is of course a random variable) by $\nu_m(n)$. The "dixie cup"-problem considered in [1] is clearly equivalent with the above problem. In [1] the mean value $\mathbf{M}(\nu_m(n))$ of $\nu_m(n)$ has been evaluated (here and in what follows $\mathbf{M}(\;)$ denotes the mean value of the random variable in the brackets) and it has been shown that

$$(1) \qquad \mathbf{M}(\nu_m(n)) = n \log n + (m-1)\, n \operatorname{loglog} n + n \cdot C_m + o(n)$$

a constant, depending on m. (The value of C_m is not given in [1]). note we shall go a step further and determine asymptoti- ution of $\nu_m(n)$; we shall prove that for every

$$\left(-\frac{e^{-x}}{(m-1)!}\right).$$

Coupon Collector's Problem

Suppose each box of cereal contains one of *N* different coupons and once a consumer has collected a coupon of each type, he can trade them for a prize. The aim of the problem is determining the average number of cereal boxes the consumer should buy to collect all coupon types, assuming that each coupon type occurs with the same probability in the cereal boxes.

Using our expected runtime calculus we showed that the expected number of necessary cereal boxes is in $\mathcal{O}(N\log(N))$.

```
$C_{\mathsf{ccp}}$:   $cp \coloneqq [0, \ldots, 0];\ i \coloneqq 1;\ x \coloneqq N;$
        while $(x > 0)$ {
            while $(cp[i] \neq 0)$ {$i :\approx \mathtt{Unif}[1 \ldots N]$};
            $cp[i] \coloneqq 1;\ x \coloneqq x-1$
        }
```

$$\mathsf{ert}\left[C_{\mathsf{ccp}}\right](\mathbf{0}) \;=\; \mathbf{4} + 2N \cdot (\mathbf{2} + \mathcal{H}_{N-1}) \in \mathcal{O}(N\log(N))$$

Summary

- Reasoning about the expected runtime of probabilistic programs a la Dijkstra
  - Handles finite and infinite runtimes
  - Establishes both bounds and exact values of the program runtimes
  - Includes several sound and complete proof rules for reasoning about loops
  - Extension with recursion has recently been provided
- Soundness of the technique w.r.t. an operational program semantics
- Extends Hoare logic for bounding the runtime of deterministic programs

Certified in Isabelle (courtesy Johannes Hölzl)

- Cases: random walk, coupon collector's problem, randomised binary search

Further details: see ESOP'16 paper; for recursion see LICS'16