Semantics of Probabilistic Programming II

Xin Zhang
Peking University

Most of the content is from "Semantics of Probabilistic Programming:
A Gentle Introduction" by Fredrik Dahlqvist, Alexandra Silva, and Dexter Kozen

Outline of the Lecture

- Syntax of a simple imperative probabilistic language
- Operational semantics
- Denotational semantics

A Simple Imperative Language

- Highly simplified version

- Enough to explain the core concepts

Syntax

- Deterministic terms (expressions)
- Terms (Deterministic + Probabilistic)
- Tests (expression that evaluate to Booleans)
- Programs

Syntax – Deterministic Terms

(i) Deterministic terms:

$$
\begin{aligned}
d ::= a \quad & \qquad a \in \mathbb{R}, \text{ constants} \\
\mid x \quad & \qquad x \in \text{Var}, \text{ a countable set of variables} \\
\mid d \ \mathbf{op} \ d \quad & \qquad \mathbf{op} \in \{+, -, *, \div\}
\end{aligned}
$$

Syntax - Terms

(ii) Terms:

$$
\begin{array}{lll}
t ::= d & & d \text{ a deterministic term} \\
\quad \mid \texttt{coin()} \mid \texttt{rand()} & & \text{sample in } \{0,1\} \text{ and } [0,1], \text{ respectively} \\
\quad \mid t \ \mathsf{op} \ t & & \mathsf{op} \in \{+,-,*,\div\}
\end{array}
$$

Syntax - Tests

(iii) Tests:

$$
\begin{array}{lll}
b ::= \texttt{true} \mid \texttt{false} & & \\
\quad \mid d == d \mid d < d \mid d > d & & \text{comparison of deterministic terms} \\
\quad \mid b \,\&\&\, b \mid b \mid\mid b \mid \texttt{!}\, b & & \text{Boolean combinations of tests}
\end{array}
$$

Syntax - Program

(iv) Programs:

$e ::= \texttt{skip}$	
$\mid x := t$	assignment
$\mid e \texttt{ ; } e$	sequential composition
$\mid \texttt{if } b \texttt{ then } e \texttt{ else } e$	conditional
$\mid \texttt{while } b \texttt{ do } e$	`while` loop

Syntax - Example Program

```
if coin() == 1 then
        x := rand() * 5
else
        x := 6
if x > 4.5 then
        y := coin() + 2
else
        y := 100
```

Operational Semantics

- Model the step-by-step executions of a program on a machine

- Tracks the memory-state
  - Values assigned to each variable
  - Values assigned to each random generator
  - A stack of instructions

Random Number Generators

- Modeled as infinite streams of numbers:
  - coin(): $m_0 m_1$ ... are i.i.d from Bernoulli(0.5)
  - rand(): $p_0 p_1$ ... are i.i.d from uniform(0, 1)

- When invoking the generator, a number is taken from the stream
  - Pseudo-random generators

Operational Semantics: Machine States

- A memory-state is a triple (s, m, p)
  - A store $s: n \to R$, where there are n variables in the program
  - $m \in \{0,1\}^{\omega}$ is the current stream of available random Boolean values
  - $p \in [0,1]^{\omega}$ is the current stream of available random real values

- A machine-state is a 4-tuple (e, s, m, p)
  - e corresponds to a stack of instructions
  - (s, m, p) is a memory-state

Machine States: Example

(e, $\{x \to \bot\}$, 1001011…, 0.2 0.5 0.9 0.21…)

if coin() == 1 then

(x := rand() * 5, $\{x \to \bot\}$, 001011…, 0.2 0.5 0.9 0.21…)

x := rand() * 5

(skip, $\{x \to 1\}$, 001011…, 0.5 0.9 0.21…)

else

x := 6

Operational Semantics: Introduction

- We now talk about how a program modifies the machine state

- Type of the operational semantics

$$(e, s, m, p) \rightarrow (e', s', m', p')$$

- Before talking about the reduction, we need to define semantics of terms and tests

Semantics of Terms

$$[\![t]\!]: \quad \boldsymbol{R}^n \times \boldsymbol{N}^\omega \times \boldsymbol{R}^\omega \to \boldsymbol{R} \times \boldsymbol{N}^\omega \times \boldsymbol{R}^\omega$$

$$[\![r]\!] : (s, m, p) \mapsto (r, m, p)$$

$$[\![x_i]\!] : (s, m, p) \mapsto (s(i), m, p)$$

$$[\![\texttt{coin()}]\!] : (s, m, p) \mapsto (\mathsf{hd}\, m, \mathsf{tl}\, m, p)$$

$$[\![\texttt{rand()}]\!] : (s, m, p) \mapsto (\mathsf{hd}\, p, m, \mathsf{tl}\, p)$$

$$[\![t_1 \ \mathsf{op} \ t_2]\!] : (s, m, p) \mapsto \begin{array}{l} \texttt{let}\ (a_1, m', p') = [\![t_1]\!](s, m, p)\ \texttt{in} \\ \texttt{let}\ (a_2, m'', p'') = [\![t_2]\!](s, m', p')\ \texttt{in} \\ (a_1\ \mathsf{op}\ a_2, m'', p'') \end{array}$$

$$opn \in \{+, 0, *, \div\}\ hd(m_1 m_2, \ldots) = m_1$$

Semantics of Tests

$$[\![b]\!]: \quad \boldsymbol{R^n \times N^\omega \times R^\omega \to \{true, false\}}$$

$$[\![t_1 == t_2]\!] : (s,m,p) \mapsto \begin{cases} \texttt{true} & \text{if } [\![t_1]\!](s,m,p) = [\![t_2]\!](s,m,p) \\ \texttt{false} & \text{otherwise} \end{cases}$$

$$[\![t_1 < t_2]\!] : (s,m,p) \mapsto \begin{cases} \texttt{true} & \text{if } [\![t_1]\!](s,m,p) < [\![t_2]\!](s,m,p) \\ \texttt{false} & \text{otherwise} \end{cases}$$

$$[\![t_1 > t_2]\!] : (s,m,p) \mapsto \begin{cases} \texttt{true} & \text{if } [\![t_1]\!](s,m,p) > [\![t_2]\!](s,m,p) \\ \texttt{false} & \text{otherwise} \end{cases}$$

$$[\![b_1 \ \&\& \ b_2]\!] : (s,m,p) \mapsto [\![b_1]\!](s,m,p) \wedge [\![b_2]\!](s,m,p)$$

$$[\![b_1 \ || \ b_2]\!] : (s,m,p) \mapsto [\![b_1]\!](s,m,p) \vee [\![b_2]\!](s,m,p)$$

$$[\![!\, b]\!] : (s,m,p) \mapsto \neg [\![b]\!](s,m,p)$$

Operational Semantics: Reduction

Assignment:

$$\frac{[\![t]\!](s,m,p) = (a,m',p')}{(x_i := t, s, m, p) \longrightarrow (\texttt{skip}, s[i \mapsto a], m', p')}$$

Sequential composition:

$$\frac{(e_1,s,m,p) \longrightarrow (e_1',s',m',p')}{(e_1 \texttt{ ; } e_2,s,m,p) \longrightarrow (e_1' \texttt{ ; } e_2,s',m',p')} \qquad \frac{}{(\texttt{skip ; } e,s,m,p) \longrightarrow (e,s,m,p)}$$

Operational Semantics: Reduction

Conditional:

$$\frac{[\![b]\!](s,m,p) = \texttt{true}}{(\texttt{if}\ b\ \texttt{then}\ e_1\ \texttt{else}\ e_2, s, m, p) \longrightarrow (e_1, s, m, p)}$$

$$\frac{[\![b]\!](s,m,p) = \texttt{false}}{(\texttt{if}\ b\ \texttt{then}\ e_1\ \texttt{else}\ e_2, s, m, p) \longrightarrow (e_2, s, m, p)}$$

`while` *loops:*

$$\frac{}{(\texttt{while}\ b\ \texttt{do}\ e, s, m, p) \longrightarrow (\texttt{if}\ b\ \texttt{then}\ (e\ ;\ \texttt{while}\ b\ \texttt{do}\ e)\ \texttt{else}\ \texttt{skip}, s, m, p)}$$

Operational Semantics: Reduction

Reflexive-transitive closure:

$$\frac{}{(e,s,m,p) \xrightarrow{*} (e,s,m,p)}$$

$$\frac{(e_1,s_1,m_1,p_1) \longrightarrow (e_2,s_2,m_2,p_2)}{(e_1,s_1,m_1,p_1) \xrightarrow{*} (e_2,s_2,m_2,p_2)}$$

$$\frac{(e_1,s_1,m_1,p_1) \xrightarrow{*} (e_2,s_2,m_2,p_2) \qquad (e_2,s_2,m_2,p_2) \xrightarrow{*} (e_3,s_3,m_3,p_3)}{(e_1,s_1,m_1,p_1) \xrightarrow{*} (e_3,s_3,m_3,p_3)}$$

Operational Semantics: Termination

- A program e terminates from (s, m, p) if

$$(e, s, m, p) \xrightarrow{*} (\texttt{skip}, s', m', p').$$

- We say e diverges from (s, m, p) if it does not terminate

Operational Semantics: Examples

x :=0
while x == 0 do
x:=coin()

What is the probability that the program halts?

$$\dfrac{\dfrac{(x := 0, s, m, p) \longrightarrow (\texttt{skip}, s[x \mapsto 0], m, p)}{(x := 0 \texttt{ ; } e, s, m, p) \longrightarrow (\texttt{skip ; } e, s[x \mapsto 0], m, p)} \quad (\texttt{skip ; } e, s[x \mapsto 0], m, p) \longrightarrow (e, s[x \mapsto 0], m, p)}{(x := 0 \texttt{ ; } e, s, m, p) \xrightarrow{*} (\texttt{skip ; } e, s[x \mapsto 0], m, p) \quad (\texttt{skip ; } e, s[x \mapsto 0], m, p) \xrightarrow{*} (e, s[x \mapsto 0], m, p)}$$

$$(x := 0 \texttt{ ; } e, s, m, p) \xrightarrow{*} (e, s[x \mapsto 0], m, p)$$

Operational Semantics: Examples

x :=0
while x == 0 **do**
x:=coin()

What is the probability that the program halts?

$$(x := 0 ; e, s, m, p) \xrightarrow{*} (e, s[x \mapsto 0], m, p)$$

$$(e, s[x \mapsto 0], m, p) \xrightarrow{*} (x := \texttt{coin()} ; e, s[x \mapsto 0], m, p)$$

$$\frac{}{(\texttt{while}\ b\ \texttt{do}\ e, s, m, p) \longrightarrow (\texttt{if}\ b\ \texttt{then}\ (e\ ;\ \texttt{while}\ b\ \texttt{do}\ e)\ \texttt{else}\ \texttt{skip}, s, m, p)}$$

$$\frac{[\![b]\!](s, m, p) = \texttt{true}}{(\texttt{if}\ b\ \texttt{then}\ e_1\ \texttt{else}\ e_2, s, m, p) \longrightarrow (e_1, s, m, p)}$$

Operational Semantics: Examples

```
x :=0
while x == 0 do
        x:=coin()
```

What is the probability that the program halts?

$$(x := 0\ ;\ e, s, m, p) \xrightarrow{*} (e, s[x \mapsto 0], m, p)$$

$$(e, s[x \mapsto 0], m, p) \xrightarrow{*} (x := \texttt{coin()}\ ;\ e, s[x \mapsto 0], m, p)$$

$$(x := \texttt{coin()}\ ;\ e, s[x \mapsto 0], m, p) \xrightarrow{*} (e, [s \mapsto \mathsf{hd}\, m], \mathsf{tl}\, m, p).$$

$$hd(m_1 m_2 \ldots) = m_1$$
$$\mathrm{tl}(m_1 m_2 \ldots) = m_2 \ldots$$

The loop continues until it reaches m inf the form of $1m'$

$$(e, s[x \mapsto 1], m', p) \xrightarrow{*} (\texttt{skip}, s[x \mapsto 1], m', p)$$

$$(x := 0\ ;\ e, s, m, p) \xrightarrow{*} (\texttt{skip}, s[x \mapsto 1], m', p)$$

Operational Semantics: Examples

$$\mathbb{P}\left[\exists m'\,(x := 0\ ;\ e, s, m, p) \xrightarrow{*} (\texttt{skip}, s[x \mapsto 1], m', p)\right]$$

$$= \mathbb{P}\left[\exists k \geq 0\ \exists m'\ m = 0^k 1 m'\right]$$

$$= \sum_{k=1}^{\infty} 2^{-k} = 1$$

Operational Semantics: Examples

```
main{
        u:=0;
        v:=0;
        step(u,v);
        while u!=0 || v!=0 do
                stepX(u,v)
}

step(u,v){
        x:=coin();
        y:=coin();
        u:=u+(x-y);
        v:=v+(x+y-1)
}
```

What is the probability that the program halts?

$$(\texttt{step}, s, 00m, p) \xrightarrow{*} (\texttt{skip}, s[(\texttt{u},\texttt{v}) \mapsto (0,-1), (\texttt{x},\texttt{y}) \mapsto (0,0)], m, p)$$

$$(\texttt{step}, s, 01m, p) \xrightarrow{*} (\texttt{skip}, s[(\texttt{u},\texttt{v}) \mapsto (-1,0), (\texttt{x},\texttt{y}) \mapsto (0,1)], m, p)$$

$$(\texttt{step}, s, 10m, p) \xrightarrow{*} (\texttt{skip}, s[(\texttt{u},\texttt{v}) \mapsto (1,0), (\texttt{x},\texttt{y}) \mapsto (1,0)], m, p)$$

$$(\texttt{step}, s, 11m, p) \xrightarrow{*} (\texttt{skip}, s[(\texttt{u},\texttt{v}) \mapsto (0,1), (\texttt{x},\texttt{y}) \mapsto (1,1)], m, p)$$

Operational Semantics: Examples

```
main{
        u:=0;
        v:=0;
        step(u,v);
        while u!=0 || v!=0 do
                stepX(u,v)
}

step(u,v){
        x:=coin();
        y:=coin();
        u:=u+(x-y);
        v:=v+(x+y-1)
}
```

What is the probability that the program halts?

We define i.i.d variables $X_1, X_2 \ldots$ on Z^2 such that

$$X_i \in \{(0,1), (0,-1), (1,0), (-1,0)\}$$

$$S_n = \sum_{i=1}^{n} X_i$$

$$(\texttt{main}, s, m, p) \xrightarrow{*}$$

$$(\texttt{while !(u == 0) || !(v == 0) do step(u, v)}, s[(\texttt{u}, \texttt{v}) \mapsto (i, j)], \mathrm{tl}^4(m), p)$$

Operational Semantics: Examples

```
main{
        u:=0;
        v:=0;
        step(u,v);
        while u!=0 || v!=0 do
                stepX(u,v)
}

step(u,v){
        x:=coin();
        y:=coin();
        u:=u+(x-y);
        v:=v+(x+y-1)
}
```

What is the probability that the program halts?

The program halts if $\exists n. S_{2n} = (0,0)$

$$(\mathtt{main}, s, m, p) \xrightarrow{*} (\mathtt{skip}, s[(\mathtt{u},\mathtt{v}) \mapsto (0,0)], \mathrm{tl}^{4n}(m), p).$$

$$\mathbb{P}\left[\exists n\ (\mathtt{main}, s, m, p) \xrightarrow{*} (\mathtt{skip}, s[(\mathtt{u},\mathtt{v}) \mapsto (0,0)], \mathrm{tl}^{4n}(m), p)\right]$$

$$= \mathbb{P}\left[\bigvee_{n=0}^{\infty} S_{2n} = (0,0)\right]$$

Operational Semantics: Examples

```
main{
        u:=0;
        v:=0;
        step(u,v);
        while u!=0 || v!=0 do
                stepX(u,v)
}

step(u,v){
        x:=coin();
        y:=coin();
        u:=u+(x-y);
        v:=v+(x+y-1)
}
```

What is the probability that the program halts?

$$\mathbb{P}[S_{2n} = (0,0)] = 4^{-2n} \sum_{m=0}^{n} \frac{(2n)!}{m!m!(n-m)!(n-m)!}$$

$$= 4^{-2n} \binom{2n}{n} \sum_{m=0}^{n} \binom{n}{m}^2$$

$$= 4^{-2n} \binom{2n}{n}^2.$$

Operational Semantics: Examples

```
i:=0;
n:=0;
while i<1e9 do
        x:=rand();
        y:=rand();
        if (x*x+y*y) < 1
                then n:=n+1;
        i:=i+1
i:=4*n/1e9;
```

Given $\epsilon > 0$, what is $\mathrm{P}(|\mathrm{i} - \pi| \leq \epsilon)$?

$$(\texttt{prog}, s, m, p) \xrightarrow{*} (\texttt{skip}, s[\texttt{i} \mapsto 4n/N, \texttt{n} \mapsto n, \ldots], m, \mathrm{tl}^{2N}(p))$$

n/N is the expectation of

$$Z = \begin{cases} 1 & \text{if } X^2 + Y^2 < 1 \\ 0 & \text{else} \end{cases}$$

Operational Semantics: Examples

```
i:=0;
n:=0;
while i<1e9 do
        x:=rand();
        y:=rand();
        if (x*x+y*y) < 1
                then n:=n+1;
        i:=i+1
i:=4*n/1e9;
```

Given $\epsilon > 0$, what is $\mathrm{P}(|\mathrm{i} - \pi| \leq \epsilon)$?

n/N is the expectation of $Z = \begin{cases} 1 & \text{if } X^2 + Y^2 < 1 \\ 0 & \text{else} \end{cases}$

$$\mathbb{P}\left[X^2 \leq t\right] = \mathbb{P}\left[X \leq \sqrt{t}\right] = \int_0^{\sqrt{t}} \mathbb{1}_{[0,1]}(x)\, dx = \sqrt{t}$$

$$f(t) = \frac{\partial \mathbb{P}\left[X^2 \leq t\right]}{\partial t} = \frac{1}{2\sqrt{t}} \mathbb{1}_{[0,1]}(t)$$

Operational Semantics: Examples

```
i:=0;
n:=0;
while i<1e9 do
        x:=rand();
        y:=rand();
        if (x*x+y*y) < 1
                then n:=n+1;
        i:=i+1
i:=4*n/1e9;
```

Given $\epsilon > 0$, what is $\mathrm{P}(|\mathrm{i} - \pi| \leq \epsilon)$?

n/N is the expectation of $Z = \begin{cases} 1 & \text{if } X^2 + Y^2 < 1 \\ 0 & \text{else} \end{cases}$

The density of $X^2 + Y^2$ is

$$(f * f)(t) = \int_{-\infty}^{\infty} \frac{1}{2\sqrt{x}} \mathbb{1}_{[0,1]}(x) \frac{1}{2\sqrt{t-x}} \mathbb{1}_{[0,1]}(t-x)\, dx$$

$$= \begin{cases} \int_0^t \frac{1}{4\sqrt{x}\sqrt{t-x}}\, dx & \text{if } 0 \leq t \leq 1 \\ \int_{t-1}^1 \frac{1}{4\sqrt{x}\sqrt{t-x}}\, dx & \text{if } 1 < t \leq 2 \end{cases}$$

Operational Semantics: Examples

```
i:=0;
n:=0;
while i<1e9 do
        x:=rand();
        y:=rand();
        if (x*x+y*y) < 1
                then n:=n+1;
        i:=i+1
i:=4*n/1e9;
```

Given $\epsilon > 0$, what is $\mathrm{P}(|\mathrm{i} - \pi| \leq \epsilon)$?

n/N is the expectation of $Z = \begin{cases} 1 & \text{if } X^2 + Y^2 < 1 \\ 0 & \text{else} \end{cases}$

The density of $X^2 + Y^2$ is

$$(f * f)(t) = \int_{-\infty}^{\infty} \frac{1}{2\sqrt{x}} \mathbb{1}_{[0,1]}(x) \frac{1}{2\sqrt{t-x}} \mathbb{1}_{[0,1]}(t-x)\, dx$$

$$= \begin{cases} \int_0^t \frac{1}{4\sqrt{x}\sqrt{t-x}}\, dx & \text{if } 0 \leq t \leq 1 \\ \int_{t-1}^1 \frac{1}{4\sqrt{x}\sqrt{t-x}}\, dx & \text{if } 1 < t \leq 2 \end{cases}$$

Operational Semantics: Examples

```
i:=0;
n:=0;
while i<1e9 do
        x:=rand();
        y:=rand();
        if (x*x+y*y) < 1
                then n:=n+1;
        i:=i+1
i:=4*n/1e9;
```

Given $\epsilon > 0$, what is $\mathrm{P}(|\mathrm{i} - \pi| \leq \epsilon)$?

n/N is the expectation of $Z = \begin{cases} 1 & \text{if } X^2 + Y^2 < 1 \\ 0 & \text{else} \end{cases}$

$\exp(Z)$ is

$$\int_0^t \frac{1}{4\sqrt{x}\sqrt{t-x}}\,dx = \int_0^1 \frac{1}{2\sqrt{1-u^2}}\,du = \frac{1}{2}(\sin^{-1}(1) - \sin^{-1}(0)) = \frac{\pi}{4}.$$

$$\mathbb{P}\left[X^2 + Y^2 \leq 1\right] = \int_0^1 (f * f)(t)\,dt = \int_0^1 \frac{\pi}{4}\,dt = \frac{\pi}{4}.$$

Operational Semantics: Examples

```
i:=0;
n:=0;
while i<1e9 do
        x:=rand();
        y:=rand();
        if (x*x+y*y) < 1
                then n:=n+1;
        i:=i+1
i:=4*n/1e9;
```

Given $\epsilon > 0$, what is $P(|i - \pi| \leq \epsilon)$?

n/N is the expectation of $Z = \begin{cases} 1 & \text{if } X^2 + Y^2 < 1 \\ 0 & \text{else} \end{cases}$

$$\mathbb{P}\left[X^2 + Y^2 \leq 1\right] = \int_0^1 (f * f)(t)\, dt = \int_0^1 \frac{\pi}{4}\, dt = \frac{\pi}{4}.$$

$$\mathbb{P}\left[\left|\frac{n}{N} - \frac{\pi}{4}\right| > \varepsilon\right] \leq \frac{\sigma^2}{N\varepsilon^2}.$$ Where $\sigma^2 = \frac{\pi}{4} - \left(\frac{\pi}{4}\right)^2$

Chebyshev's inequality

Denotational vs. Operational Semantics

- Consider x := coin(), in operational semantics:

$$(\texttt{x} := \texttt{coin()}, s, m, p) \longrightarrow (\texttt{skip}, s[\texttt{x} \mapsto \texttt{0}], \mathsf{tl}\, m, p)$$

$$(\texttt{x} := \texttt{coin()}, s, m, p) \longrightarrow (\texttt{skip}, s[\texttt{x} \mapsto \texttt{1}], \mathsf{tl}\, m, p)$$

- Denotational semantics:
  - Model all possible executions together
  - States: probability distribution over memory states
  - $\frac{1}{2}s[x \mapsto 0] + \frac{1}{2}s[x \mapsto 1]$

Denotational Semantics: Introduction

- State s can be identified with the Dirac measure σ_s, then the semantics of x:=coin() can be viewed as $\sigma_s \to \frac{1}{2}s[x \mapsto 0] + \frac{1}{2}s[x \mapsto 1]$

- In general, a program is interpreted as an operator mapping probability distributions to (sub)probability distributions

Denotational Semantics: Definition

- Assume there are n real variables, then a state is a distribution on R^n

- A program $e: MR^n \to MR^n$
  - An operator called a state transformer

Denotational Semantics: Terms

$$[\![r]\!](a_1,\ldots,a_n) = \delta_r,\ r \in \mathbb{R}$$

$$[\![x_i]\!](a_1,\ldots,a_n) = \delta_{a_i}$$

$$[\![\texttt{coin()}]\!](a_1,\ldots,a_n) = \frac{1}{2}\delta_0 + \frac{1}{2}\delta_1$$

$$[\![\texttt{rand()}]\!](a_1,\ldots,a_n) = \lambda$$

$$[\![t_1\ \mathrm{op}\ t_2]\!](a_1,\ldots,a_n) = \mathrm{op}_*([\![t_1]\!](a_1,\ldots,a_n) \otimes [\![t_2]\!](a_1,\ldots,a_n))$$

The denotational semantics of any term is a Markov kernel $\boldsymbol{R^n \to MR}$

$op \in \{+,-,\times,\div\}, \lambda$ *is the Lebesgue measure on* $[0,1]$, op_* *denotes push forward*

$$\mathrm{op}_*([\![t_1]\!](a_1,\ldots,a_n) \otimes [\![t_2]\!](a_1,\ldots,a_n))(B)$$

$$= [\![t_1]\!](a_1,\ldots,a_n) \otimes [\![t_2]\!](a_1,\ldots,a_n)\left\{(u,v) \in \mathbb{R}^2 \mid u\ \mathrm{op}\ v \in B\right\}.$$

Denotational Semantics: Tests

$$[\![t_1 == t_2]\!] = \{(a_1, \ldots, a_n) \in \mathbb{R}^n \mid [\![t_1]\!](a_1, \ldots, a_n) = [\![t_2]\!](a_1, \ldots, a_n)\}$$

$$[\![t_1 < t_2]\!] = \{(a_1, \ldots, a_n) \in \mathbb{R}^n \mid [\![t_1]\!](a_1, \ldots, a_n) < [\![t_2]\!](a_1, \ldots, a_n)\}$$

$$[\![t_1 > t_2]\!] = \{(a_1, \ldots, a_n) \in \mathbb{R}^n \mid [\![t_1]\!](a_1, \ldots, a_n) > [\![t_2]\!](a_1, \ldots, a_n)\}$$

$$[\![b_1 \;\&\&\; b_2]\!] = [\![b_1]\!] \cap [\![b_2]\!]$$

$$[\![b_1 \;||\; b_2]\!] = [\![b_1]\!] \cup [\![b_2]\!]$$

$$[\![!b]\!] = [\![b]\!]^c.$$

The denotational semantics of any test is a measurable subset of R^n

Denotational Semantics: Tests

- How are semantics of tests used?

Each measurable set $B \subseteq R^n$ defines a linear operator:

$$T_B(\mu)(C) = \mu(B \cap C).$$

- Tests usually send states to sub-probability distributions

Denotational Semantics: Program

Programs are interpreted as operators $\mathcal{M}\mathbb{R}^n \to \mathcal{M}\mathbb{R}^n$

(i) $[\![\texttt{skip}]\!] = \mathrm{Id}_{\mathcal{M}\mathbb{R}^n} : \mathcal{M}\mathbb{R}^n \to \mathcal{M}\mathbb{R}^n$, the identity operator $\mu \mapsto \mu$.

(ii) Assignments:

Given such a term t and an index $1 \leq i \leq n$, we define the Markov kernel $F_t^i : \mathbb{R}^n \to \mathcal{M}\mathbb{R}^n$ as

$$F_t^i(x_1, \ldots, x_n) = \delta_{x_1} \otimes \ldots \otimes \delta_{i-1} \otimes [\![t]\!](x_1, \ldots, x_n) \otimes \delta_{x_{i+1}} \otimes \ldots \otimes \delta_{x_n}$$

Using this definition, we define $[\![x_i := t]\!]$ as the operator

$$[\![x_i := t]\!](\mu) = (F_t^i)_*(\mu)$$ (F_t^i) is the push forward of the Markov kernel

Denotational Semantics: More on Assignments

$$
\begin{aligned}
&[\![x_i := r]\!](\mu)(B_1 \times \cdots \times B_n) \\
&\quad = \int_{\mathbb{R}^n} \delta_{x_1} \otimes \cdots \otimes \delta_{x_{i-1}} \otimes \delta_r \otimes \delta_{x_{i+1}} \otimes \cdots \otimes \delta_{x_n}(B_1 \otimes \cdots \otimes B_n)\, d\mu \\
&\quad = \int_{\mathbb{R}^n} \delta_{x_1}(B_1) \cdots \delta_{x_{i-1}}(B_{i-1}) \delta_r(B_i) \delta_{x_{i+1}}(B_{i+1}) \cdots \delta_{x_n}(B_n)\, d\mu \\
&\quad = \mu(B_1 \times \cdots \times B_{i-1} \times \mathbb{R} \times B_{i+1} \times \cdots \times B_n) \delta_r(B_i).
\end{aligned}
$$

$$
\begin{aligned}
&[\![x_i := \texttt{coin()}]\!](\mu)(B_1 \times \cdots \times B_n) \\
&\quad = \mu(B_1 \times \cdots \times B_{i-1} \times \mathbb{R} \times B_{i+1} \times \cdots \times B_n) \left(\frac{1}{2}\delta_0 + \frac{1}{2}\delta_1\right)(B_i).
\end{aligned}
$$

Denotational Semantics: Program

(iii) $[\![e_1 \texttt{ ; } e_2]\!] = [\![e_2]\!] \circ [\![e_1]\!]$, the usual composition of operators.

(iv) $\texttt{if } b \texttt{ then } e_1 \texttt{ else } e_2$ is the operator defined by

$$[\![\texttt{if } b \texttt{ then } e_1 \texttt{ else } e_2]\!] = [\![e_1]\!] \circ T_{[\![b]\!]} + [\![e_2]\!] \circ T_{[\![b]\!]^c}$$

Denotational Semantics: Program

(v) To define the semantics of `while` loops, we use the equivalence of the following two programs,

$$\texttt{while}\ b\ \texttt{do}\ e \qquad\qquad \texttt{if}\ b\ \texttt{then}\ (e\ ;\ \texttt{while}\ b\ \texttt{do}\ e)\ \texttt{else skip},$$

$$[\![\texttt{while}\ b\ \texttt{do}\ e]\!] = [\![\texttt{while}\ b\ \texttt{do}\ e]\!] \circ [\![e]\!] \circ T_{[\![b]\!]} + T_{[\![b]\!]^c}$$

Which is the least fixed point of $\tau(S) = S \circ [\![e]\!] \circ T_{[\![b]\!]} + T_{[\![b]\!]^c}$

$$[\![\texttt{while}\ b\ \texttt{do}\ e]\!] = \bigvee_{n\geq 0} \tau^n(0) = \sum_{k=0}^{\infty} T_{[\![b]\!]^c} \circ \left([\![e]\!] \circ T_{[\![b]\!]}\right)^k$$

Constructable because norm-bounded directed sets have suprema and that τ preserves such suprema

Denotational Semantics: Program

Theorem 1.5 *The denotational semantics of any program given by the syntax of Section 1.3.1 is a positive operator of norm at most one.*

Denotational Semantics: Examples

$$[\![\texttt{x := 0 ; while x == 0 do x := coin()}]\!] : \mathcal{M}\mathbb{R} \to \mathcal{M}\mathbb{R}$$

$$[\![\texttt{x := 0}]\!] : \mathcal{M}\mathbb{R} \to \mathcal{M}\mathbb{R} \qquad \mu \mapsto \mu(\mathbb{R})\delta_0,$$

$$[\![\texttt{x := coin()}]\!] : \mathcal{M}\mathbb{R} \to \mathcal{M}\mathbb{R} \qquad \mu \mapsto \mu(\mathbb{R})(\tfrac{1}{2}\delta_0 + \tfrac{1}{2}\delta_1)$$

What about the while loop?

Denotational Semantics: Examples

$$[\![\texttt{x := 0 ; while x == 0 do x := coin()}]\!] : \mathcal{M}\mathbb{R} \to \mathcal{M}\mathbb{R}$$

$$\tau^0(0)(\mu) = 0$$

$$\tau^1(0)(\mu) = T_{\{0\}^c}(\mu) = \mu(- \cap \{0\}^c)$$

$$\tau^2(0)(\mu) = \left(T_{\{0\}^c} + T_{\{0\}^c} \circ [\![\texttt{x := coin()}]\!] \circ T_{\{0\}}\right)(\mu)$$
$$= \mu(- \cap \{0\}^c) + \frac{\mu(\{0\})}{2}\delta_1.$$

$$\tau^k(0)(\mu) = \mu(- \cap \{0\}^c) + \sum_{i=1}^{k-1} \frac{\mu(0)}{2^i}\delta_1$$
$$= \mu(- \cap \{0\}^c) + (1 - 2^{-(k-1)})\mu(\{0\})\delta_1,$$

It's limit is $\mu(- \cap \{0\}^c) + \mu(\{0\})\delta_1$

$$[\![\texttt{x := 0 ; while x == 0 do x := coin()}]\!] : \mathcal{M}\mathbb{R} \to \mathcal{M}\mathbb{R} \qquad \mu \mapsto \mu(\mathbb{R})\delta_1.$$

More Examples

- Refer to the supplementary reading
- https://www.cambridge.org/core/services/aop-cambridge-core/content/view/A7964205E44B5234A78C661192E294E1/9781108488518c1_1-42.pdf/semantics_of_probabilistic_programming_a_gentle_introduction.pdf

Left Issues: Initial State in Denotational

- We can assume that each variable is assigned to a default value

- A Dirac measure

Left Issues: Condition

- Operational semantics: model the traces that fails to satisfies the condition as erroneous traces

- Denotational semantics: similar to tests, but need to renormalizes the measures

Left Issues: Non-Terminating Executions

$$\texttt{while true do } e.$$

- Operational semantics: infinite trace

- Denotational:
  - Can declare infinite traces are invalid
  - Can still compute some least fixed point (more involved)

Next Lecture

- Inference in probabilistic programs
  - Instantiations of (sampling) methods we have studied

Mid-Term

- Content: lecture 1-5 (semantics not included)
- Open book
  - Complex formulae will be provided. For example: detailed balance will be on the exam paper, but not the definition of stationary distribution
  - You can print whatever you want on two sheets of A4 or letter-size papers (4 pages)
  - No electronic devises
- Respect academic integrity
- Online attendees: we will talk offline