Semantics of Probabilistic Programming

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
Recap: Problem and Motivation

• Evaluate $P(Z|X)$ and related expectations

• Problem with exact methods
  • Curse of dimensionality

• $P(Z|X)$ has a complex form making expectations analytically intractable
Recap: Variational Inference

• Functional: a function that maps a function to a value

$$H[p] = \int p(x) \ln p(x) \, dx$$

• Variational method: find an input function that maximizes the functional

• Variational inference: find a distribution $q(z)$ to approximate $p(Z \mid X)$ so a functional is maximized
Recap: Variational Inference

\[ \ln p(X) = \mathcal{L}(q) + \text{KL}(q||p) \]

\[ \mathcal{L}(q) = \int q(Z) \ln \left\{ \frac{p(X, Z)}{q(Z)} \right\} \, dZ \]

\[ \text{KL}(q||p) = -\int q(Z) \ln \left\{ \frac{p(Z|X)}{q(Z)} \right\} \, dZ \]

If \( q \) can be any distribution, then variational inference is precise. But in practice, it cannot.
Is the following statement right?

• Probability $p(Z,X)$ is usually easier to evaluate compared to $P(Z | X)$. 
Recap: Sampling Methods

• Stochastic methods

• Also called Monte Carlo methods

\[
E[f] = \int f(z)p(z) \, dz \quad \Rightarrow \quad \hat{f} = \frac{1}{L} \sum_{l=1}^{L} f(z^{(l)}) \quad z_1, \ldots, z_l \text{ are samples from } p
\]
Recap: Sampling Methods

• Transformation method: CDF\(^{-1}\)(uniform(0,1))

• Rejection sampling
  • A proposal distribution q(z)
  • Choose k, such that k*q(z) \(\geq\) p(z), for any x
  • Sampling process:
    • Sample \(z_0\) from q(z)
    • Sample h from uniform(0, k*q(\(z_0\)))
    • If h > p(\(z_0\)), discard it; otherwise, keep it
Is the following statement correct?

• All primitive distributions can be constructed using the transformation method.
Is the following statement right?

• In rejection sampling, given k, the probability whether a sample is accepted does not depend on the proposal distribution.
Is the following statement correct?

• The efficiency of rejection sampling depends on the choice of the proposal distribution
Recap: Sampling Methods

• Importance sampling
  • Used to evaluate $f(z)$ where $z$ is from $p(z)$

$$E(f) = \int f(z)p(z)dz = \int f(z)\frac{p(z)}{q(z)}q(z)dz \approx \frac{1}{L} \sum_{l=1}^{L} \frac{p(z^l)}{q(z^l)}f(z^l)$$

• How to get real samples: create a new discrete distribution using the above samples and set their probabilities using the importance weights
Recap: Sampling Methods

• Markov Chain Monte Carlo
  • A sampling method that works with a large family of distributions and high dimensions

• Workflow
  • Start with some sample $z_0$
  • Suppose the current sample is $z^\tau$. Draw next sample $z^*$ from $q(z | z^\tau)$
  • Decide whether to accept $z^*$ as the next state based some criteria. If accepted, $z^{\tau+1} = z^*$. Otherwise, $z^{\tau+1} = z^\tau$
  • Samples form a Markov chain
Recap: Sampling Methods

<table>
<thead>
<tr>
<th>Constraints on the proposal distribution</th>
<th>Metropolis</th>
<th>Metropolis-Hasting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symmetric</td>
<td></td>
<td>None</td>
</tr>
</tbody>
</table>

Accepting probability

\[
\begin{align*}
\text{Metropolis: } & \min(1, \frac{p(z')}{p(z)}) \\
\text{Metropolis-Hasting: } & \min(1, \frac{p(z')q(z'|z)}{p(z)q(z'|z')})
\end{align*}
\]
Recap: Why MCMC works?

• Markov chain: 
  \[ p(z^{(m+1)}|z^{(1)}, \ldots, z^{(m)}) = p(z^{(m+1)}|z^{(m)}) \].

• Stationary distribution of a Markov chain: each step in the chain does not change the distribution.

  • Detailed balance: 
    \[ p^*(z)T(z, z') = p^*(z')T(z', z) \]
    • \( p^*(z) \) is a stationary distribution

  • A *ergodic* Markov chain converges to the same distribution regardless the initial distribution
    • The system does not return to the same state at fixed intervals
    • The expected number of steps for returning to the same state is finite
Is the following statement right?

• The samples drawn using MCMC are independent
Is the following statement right?

• A Markov chain can have more than one stationary distribution
Use MCMC to solve the problem below

• Super optimization
  • There is a straight-line program
  • A set of test cases are given
  • The program can be modified by deleting a statement, inserting a statement from the initial program at a given place
  • Optimize the program by using the above operations
Motivations

• In order to reason about properties of a program, we need formal tools

• Example questions
  • Is the postcondition satisfied?
  • Does this program halt on all inputs?
  • Does it always halt in polynomial time?
Motivations

• In order to reason about properties of a program, we need formal tools.

• Example questions
  • What is the probability that the postcondition is satisfied?
  • What is the probability that this program halts on all inputs?
  • What is the probability that it halts in polynomial time?
Motivations

• When designing a language, rigorous semantics is needed to guarantee its correctness

• An example that didn’t have rigorous semantics: Javascript
  • https://javascriptwtf.com
Examples

\[ x := 0 \]

\[ \text{while } x == 0 \text{ do} \]
\[ x := \text{coin()} \]

What is the probability that it runs through \( n \) iterations?
What is the expected number of iterations?
What is the probability that the program halts?

We can decompose the semantics of a program into semantics of statements.
Examples

```
main{
    u:=0;
    v:=0;
    step(u,v);
    while u!=0 || v!=0 do
        step(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 is a two-dimensional random walk. According to probability theory, the probability that it returns to the origin is 1.

By relating to concepts in probabilities, we can simplify the reasoning...
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;

What does this program compute?

How to reason about it?

Measure Theory
The mathematical foundation of probabilities and integration

Uniform(0,1) is called a *Lebesgue measure*
This Class

• Syntax of a simple imperative probabilistic language

• Operational semantics

• Measure theory & denotational semantics (brief)
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:

\[ d ::= a \quad a \in \mathbb{R}, \text{ constants} \]
\[ | x \quad x \in \text{Var}, \text{ a countable set of variables} \]
\[ | d \text{ op } d \quad \text{op} \in \{+,-,\ast,\div\} \]
Syntax - Terms

(ii) Terms:

\[ t ::= d \quad d \text{ a deterministic term} \]
\[ \mid \text{coin()} \mid \text{rand()} \quad \text{sample in } \{0, 1\} \text{ and } [0, 1], \text{respectively} \]
\[ \mid t \text{ op } t \quad \text{op } \in \{+, -, \ast, \div\} \]
(iii) Tests:

\[
\begin{align*}
    b & := \text{true} \mid \text{false} \\
    & \mid d == d \mid d < d \mid d > d \\
    & \mid b \&\& b \mid b \mid b \mid !b
\end{align*}
\]

- comparison of deterministic terms
- Boolean combinations of tests
Syntax - Program

(iv) Programs:

\[ e ::= \text{skip} \]
\[ \mid x := t \quad \text{assignment} \]
\[ \mid e ; e \quad \text{sequential composition} \]
\[ \mid \text{if } b \text{ then } e \text{ else } e \quad \text{conditional} \]
\[ \mid \text{while } b \text{ do } e \quad \text{while loop} \]
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 of each random number 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 \(s: n \rightarrow R\), 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

\[(e, \{x \rightarrow \bot\}, 1001011..., 0.2 \ 0.5 \ 0.9 \ 0.21...)\]

if \(\text{coin()} == 1\) then
\[(x := \text{rand()} \ast 5, \{x \rightarrow \bot\}, 001011..., 0.2 \ 0.5 \ 0.9 \ 0.21...)\]
\[x := \text{rand()} \ast 5\]
\[(\text{skip}, \{x \rightarrow 1\}, 001011..., 0.5 \ 0.9 \ 0.21...)\]
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

\[ [t] : \quad R^n \times N^\omega \times R^\omega \rightarrow R \times N^\omega \times R^\omega \]

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

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

\[ [\text{coin()}] : (s, m, p) \mapsto (\text{hd } m, \text{tl } m, p) \]

\[ [\text{rand()}] : (s, m, p) \mapsto (\text{hd } p, m, \text{tl } p) \]

\[ [t_1 \text{ op } t_2] : (s, m, p) \mapsto \text{ let } (a_1, m', p') = [t_1](s, m, p) \text{ in } \]
\[ \text{ let } (a_2, m'', p'') = [t_2](s, m', p') \text{ in } (a_1 \text{ op } a_2, m'', p'') \]

\[ \text{opn } \in \{+, 0, *, \div\} \text{ hd}(m_1m_2, ...) = m_1 \]
Semantics of Tests

\[
\llbracket b \rrbracket : \quad R^n \times N^\omega \times R^\omega \rightarrow \{true, false\}
\]

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

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

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

\[
\llbracket b_1 \& b_2 \rrbracket : (s, m, p) \mapsto \llbracket b_1 \rrbracket(s, m, p) \land \llbracket b_2 \rrbracket(s, m, p)
\]

\[
\llbracket b_1 \lor b_2 \rrbracket : (s, m, p) \mapsto \llbracket b_1 \rrbracket(s, m, p) \lor \llbracket b_2 \rrbracket(s, m, p)
\]

\[
\llbracket \neg b \rrbracket : (s, m, p) \mapsto \neg \llbracket b \rrbracket(s, m, p)
\]
Operational Semantics: Reduction

Assignment:

\[
[[t]](s, m, p) = (a, m', p')
\]

\[
(x_i := t, s, m, p) \rightarrow (\text{skip}, s[i \leftarrow a], m', p')
\]

Sequential composition:

\[
(e_1, s, m, p) \rightarrow (e'_1, s', m', p')
\]

\[
(e_1 ; e_2, s, m, p) \rightarrow (e'_1 ; e_2, s', m', p')
\]

\[
(\text{skip} ; e, s, m, p) \rightarrow (e, s, m, p)
\]
Operational Semantics: Reduction

*Conditional:*

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

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

*while loops:*

\[
(\text{while } b \text{ do } e, s, m, p) \longrightarrow (\text{if } b \text{ then } (e \text{ ; while } b \text{ do } e) \text{ else skip}, s, m, p)
\]
Operational Semantics: Reduction

Reflexive-transitive closure:

\[(e, s, m, p) \rightarrow^* (e, s, m, p)\]
\[(e_1, s_1, m_1, p_1) \rightarrow (e_2, s_2, m_2, p_2)\]
\[(e_1, s_1, m_1, p_1) \rightarrow^* (e_2, s_2, m_2, p_2)\]

\[(e_1, s_1, m_1, p_1) \rightarrow (e_2, s_2, m_2, p_2)\]
\[(e_2, s_2, m_2, p_2) \rightarrow^* (e_3, s_3, m_3, p_3)\]

\[(e_1, s_1, m_1, p_1) \rightarrow (e_3, s_3, m_3, p_3)\]
Operational Semantics: Termination

- A program $e$ terminates from $(s, m, p)$ if
  $$(e, s, m, p) \xrightarrow{*} (\text{skip}, s', m', p').$$

- We say $e$ diverges from $(s, m, p)$ if it does not terminate.
Operational Semantics: Examples

\[ x := 0 \]
\[ \text{while } x == 0 \text{ do} \]
\[ x := \text{coin()} \]

What is the probability that the program halts?

\[
\begin{align*}
(x := 0, s, m, p) &\rightarrow (\text{skip}, s[x \mapsto 0], m, p) \\
(x := 0; e, s, m, p) &\rightarrow (\text{skip}; e, s[x \mapsto 0], m, p) \\
(x := 0; e, s, m, p) &\rightarrow^* (\text{skip}; e, s[x \mapsto 0], m, p) \\
(skip; e, s[x \mapsto 0], m, p) &\rightarrow (e, s[x \mapsto 0], m, p) \\
(skip; e, s[x \mapsto 0], m, p) &\rightarrow^* (e, s[x \mapsto 0], m, p) \\
(x := 0; e, s, m, p) &\rightarrow^* (e, s[x \mapsto 0], m, p)
\end{align*}
\]
Operational Semantics: Examples

\[
x := 0 \\
\text{while } x == 0 \text{ do} \\
x := \text{coin()}
\]

What is the probability that the program halts?

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

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

\[
(\text{while } b \text{ do } e, s, m, p) \rightarrow (\text{if } b \text{ then } (e; \text{while } b \text{ do } e) \text{ else skip}, s, m, p)
\]

\[
[b](s, m, p) = \text{true} \quad \Rightarrow \quad (\text{if } b \text{ then } e_1 \text{ else } e_2, s, m, p) \rightarrow (e_1, s, m, p)
\]
Operational Semantics: Examples

\[
x := 0 \\
\textbf{while} \ x == 0 \ \textbf{do} \\
x := \text{coin()}
\]

What is the probability that the program halts?

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

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

The loop continues until it reaches \(m\) in the form of \(1m'\)

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

\[
(x := 0 \ ; \ e, s, m, p) \xrightarrow{*} (\text{skip}, s[x \mapsto 1], m', p)
\]
Operational Semantics: Examples

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

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

\[ = \sum_{k=1}^{\infty} 2^{-k} = 1 \]
Operational Semantics: Examples

What is the probability that the program halts?

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

```
step(u,v){
  x:=coin();
  y:=coin();
  u:=u+(x-y);
  v:=v+(x+y-1)
}
```
main{
    u:=0;
    v:=0;
    step(u,v);
    while u!=0 || v!=0 do
        step(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 $\mathbb{Z}^2$ such that $X_i \in \{(0,1), (0,-1), (1,0), (-1,0)\}$

$$S_n = \sum_{i=1}^{n} X_i$$
Operational Semantics: Examples

What is the probability that the program halts?

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

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

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

$$= \mathbb{P} \left[ \bigvee_{n=0}^{\infty} S_{2n} = (0, 0) \right]$$
main{
    u:=0;
    v:=0;
    step(u,v);
    while u!=0 || v!=0 do
        step(u,v)
}

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

\[
i := 0; \\
n := 0; \\
\text{while } i < 1e9 \text{ do} \\
\quad x := \text{rand}(); \\
\quad y := \text{rand}(); \\
\quad \text{if } (x \times x + y \times y) < 1 \\
\qquad \text{then } n := n + 1; \\
\quad i := i + 1 \\
i := 4 \times n / 1e9;
\]

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

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

$n/N$ is the expectation of

\[
Z = \begin{cases} 
1 & \text{if } X^2 + Y^2 < 1 \\
0 & \text{else}
\end{cases}
\]
Given $\epsilon > 0$, what is $P(|i - \pi| \leq \epsilon)$?

$n/N$ is the expectation of

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

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

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

\begin{align*}
i &:= 0; \\
n &:= 0; \\
\text{while } i < 1e9 \text{ do} \\
& \quad \begin{align*}
x &:= \text{rand}(); \\
y &:= \text{rand}(); \\
\text{if } (x^2 + y^2) < 1 & \quad \text{then } n := n + 1; \\
\end{align*} \\
i &:= i + 1 \\
i &:= 4*n/1e9;
\end{align*}

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

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

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

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

\[
= \begin{cases} 
\int_{0}^{t} \frac{1}{4\sqrt{x}\sqrt{t-x}} \, dx & \text{if } 0 \leq t \leq 1 \\
\int_{0}^{1} \frac{1}{4\sqrt{x}\sqrt{t-x}} \, dx & \text{if } 1 < t \leq 2
\end{cases}
\]
Operational Semantics: Examples

\[ i := 0; \]
\[ n := 0; \]
\[ \text{while } i < 1e9 \text{ do} \]
\[ x := \text{rand}(); \]
\[ y := \text{rand}(); \]
\[ \text{if } (x^2 + y^2) < 1 \]
\[ \text{then } n := n + 1; \]
\[ i := i + 1 \]
\[ i := 4n / 1e9; \]

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

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

The density of \( X^2 + Y^2 \) is

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

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

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

exp$(Z)$ is  

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

$$P[X^2 + Y^2 \leq 1] = \int_0^1 (f * f)(t) \, dt = \int_0^1 \frac{\pi}{4} \, dt = \frac{\pi}{4}.$$
Operational Semantics: Examples

\begin{align*}
i &:= 0; \\
n &:= 0; \\
\text{while } i < 1e9 \text{ do} & \\
& \quad \text{\hspace{1cm} } x := \text{rand}(); \\
& \quad \text{\hspace{1cm} } y := \text{rand}(); \\
& \quad \text{\hspace{1cm} } \text{if } (x^2 + y^2) < 1 \\
& \quad \text{\hspace{1.5cm} } \text{then } n := n + 1; \\
& \quad \text{\hspace{1cm} } i := i + 1 \\
i &:= 4n / 1e9;
\end{align*}

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

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

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

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

Chebyshev’s inequality
This Class

• Syntax of a simple imperative probabilistic language

• Operational semantics

• Measure theory & denotational semantics (brief)
Denotational vs. Operational Semantics

• Consider \( x := \text{coin}() \), in operational semantics:

\[
(x := \text{coin}(), s, m, p) \longrightarrow (\text{skip}, s[x \mapsto 0], \text{tl} m, p)
\]
\[
(x := \text{coin}(), s, m, p) \longrightarrow (\text{skip}, s[x \mapsto 1], \text{tl} m, p)
\]

• 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 $\sigma_s$, then the semantics of $x := \text{coin()}$ can be viewed as $\sigma_s \rightarrow \frac{1}{2} s[x \mapsto 0] + \frac{1}{2} s[x \mapsto 1]$

• 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 $e: MR^n \rightarrow MR^n$
  • An operator called a state transformer
Measure Theory

• Measures: generalization of concepts like length, area, or volume
Measure Example: Length

• What subsets of $\mathbb{R}$ can meaningfully be assigned a length?

• What properties should the length function $l$ satisfy?
Measure Example: Length

\[ \ell([a_1, b_1] \cup [a_2, b_2]) = \ell([a_1, b_1]) + \ell([a_2, b_2]) = (b_1 - a_1) + (b_2 - a_2). \]

\[ b_1 < a_2 \]

\[ \ell \left( \bigcup_{i=1}^{n} A_i \right) = \sum_{i=1}^{n} \ell(A_i). \quad A_i \text{ and } A_j \text{ are disjoined } \]

\[ \ell \left( \bigcup_{i=0}^{\infty} A_i \right) = \sum_{i=0}^{\infty} \ell(A_i). \quad A_i \text{ and } A_j \text{ are disjoined } \]

\[ \text{The set is countable. } \ell \text{ is called countably additive or } \sigma - \text{additive} \]

\[ \ell(R) = \infty, \text{ but we are only going to talk about finite measures} \]

\[ \ell(B \setminus A) = \ell(B) - \ell(A) \quad \text{Domain should be closed under complementation} \]
Measure Example: Length

• Can we extend the domain of length $l$ to all subsets of $\mathbb{R}$?

• No. Counterexample: Vitali sets
  • $V \subseteq [0,1]$, such that for each real number $r$, there exists exactly one number $v \in V$ such that $v - r$ is rational
  • Let $q_1, q_2, \ldots$ be the rational numbers in $[-1,1]$, construct sets $V_k = V + q_k$
  • $[0,1] \subseteq \bigcup V_k \subseteq [-1,2]$
  • $l(V_k) = l(V)$, and there are infinitely many $V_k$

• $l$ is called the Lebesgue measure on real numbers
Measurable Spaces and Measures

• \((S, \mathcal{B})\) is a measurable space
  • \(S\) is a set
  • \(\mathcal{B}\) is a \(\sigma\)-algebra on \(S\), which is a collection of subsets of \(S\)
    • It contains \(\emptyset\)
    • Closed under complementation in \(S\)
    • Closed under countable union
  • The elements of \(\mathcal{B}\) are called measurable sets

• If \(F\) is a collection of subsets of \(S\), \(\sigma(F)\) is the smallest \(\sigma\)-algebra containing \(F\), or \(\sigma(F) \triangleq \bigcap \{\mathcal{A} \mid F \subseteq \mathcal{A} \text{ and } \mathcal{A} \text{ is a } \sigma\text{-algebra}\}\). We say \((S, \sigma(F))\) is generated by \(F\).
Measurable Functions

• $(S, \mathcal{B}_S)$ and $(T, \mathcal{B}_T)$ are measurable spaces. A function $f: S \to T$ is measurable if $f^{-1}(B) = \{x \in S | f(x) \in B\}$ for every $B \in \mathcal{B}_T$ is a measurable subset of $S$.

Example:

$$\chi_B(s) = \begin{cases} 1, & s \in B, \\ 0, & s \notin B. \end{cases}$$
Measures: Definitions

• A signed (finite) measure on \((S, B)\) is a countably additive map \(\mu: B \rightarrow \mathbb{R}\) such that \(\mu(\emptyset) = 0\)

• Positive signed measure: \(\mu(A) \geq 0\) for all \(A \in B\)

• A positive measure is a probability measure if \(\mu(S) = 1\)

• …is a subprobability measure if \(\mu(S) \leq 1\)
Measures: Definitions

• If $\mu(B) = 0$, then $B$ is a $\mu$-nullset

• A property is said to hold $\mu$-almost surely (everywhere) if the sets of points on which it does not hold is contained in nullset

• In probability theory, measures are sometimes called distributions
Measures: Discrete Measures

• For \( s \in S \), the Dirac measure, or Dirac delta, or point mass on \( s \):

\[
\delta_s(B) = \begin{cases} 
1, & s \in B, \\
0, & s \notin B.
\end{cases}
\]

• A measure is discrete if it is a countable weighted sum of Dirac measures
  • If the weights add up to one, then it is a discrete probability measure

• Continuous measure: \( \mu(\{s\}) = 0 \) for all singleton sets \( \{s\} \) in \( B \) of \( (S, B) \)
Measures: Pushforward Measure and Lebesgue Integration

• Given $f: (S, B_S) \rightarrow (T, B_T)$ measurable, an a measure $\mu$ on $B_S$, the
  pushfoward measure $\mu(f^{-1}(B))$ on $B_T$ is defined as

$$f_*(\mu)(B) = \mu(f^{-1}(B)), \quad B \in B_T.$$ 

• Lebesgue integration: given $(S, B), \mu: B \rightarrow \mathbb{R}, f: S \rightarrow \mathbb{R}$, where $m < f < M$

$$\int f \, d\mu = \lim_{n \rightarrow \infty} \sum_{i=0}^{n} f(s_i)\mu(B_i)$$

where $B_0, \ldots, B_n$ is a measurable partition of $S$, and the value of $f$ does
not vary more than $(M - m)/n$ in any $B_i$ and $s_i \in B_i$
Markov Kernels

• Given \((S, B_S)\) and \((T, B_T)\), \(P: S \times B_T \to \mathbb{R}\) is called a Markov kernel if
  • For fixed \(A \in B_T\), the map \(\lambda s. P(s, A) \to \mathbb{R}\) is a measurable function on \((S, B_S)\)
  • For fixed \(s \in S\), the map \(\lambda A. P(s, A) \to \mathbb{R}\) is a probability measure on \((T, B_T)\)

• Composition of two Markov kernels
  • Given \(P: S \to T\), \(Q: T \to U\) \((P ; Q)(s, A) = \int_{t \in T} P(s, dt) \cdot Q(t, A)\).

• Given \(\mu\) on \(B_S\), its push forward under the Markov Kernel \(P\) is

\[
P_\ast(\mu)(B) = \int_{s \in S} P(s, B) \mu(ds).
\]
More on Markov Kernels

• \((S, B_S)\): \(x = \ldots \) (\(x > 0\))

• \((T, B_T)\): \(y = \text{uniform}(0, x)\)

• Markov kernel \(P(x, \bigcup_{i=1}^{M} [a_i, b_i]) = \sum_{i=1}^{M} \text{length}([a_i, b_i] \cap [0, x])/x\)
More on Markov Kernels

• \((S, B_S)\): \(x = \ldots (x > 0)\)

• \((T, B_T)\): \(y = \text{uniform}(0, x)\)

• \((T, B_T)\): \(z = \text{uniform}(0, y)\)

• Composition: \((P; Q)(x, [0, z]) = \int_{y \in [0, \infty]} P(x, dy) \ast Q(y, [0, z])\)
  
  \[
  z < x
  \]

  \[
  = \int_{y \in [0, x]} \frac{dy}{x} \ast \frac{y}{\text{length}([0, z] \cap [0, y])}
  \]

  \[
  = \int_{y \in [0, z]} \frac{dy}{x} \ast \frac{y}{y} + \int_{y \in [z, x]} \frac{dy}{x} \ast \frac{z}{y} = \frac{z}{x} + \frac{z}{x} (\ln x - \ln z)
  \]
More on Markov Kernels

\(\mathbf{S}, \mathcal{B}_\mathbf{S}\): \(x = \text{uniform}(0.1, 1.1)\) \(\mu([a, b]) = \text{length}([a, b] \cap [0.1, 1.1])\)

\(\mathbf{T}, \mathcal{B}_\mathbf{T}\): \(y = \text{uniform}(0, x)\)

Markov kernel \(P(x, \bigcup_{i=1}^{M} [a_i, b_i]) = \sum_{i=1}^{M} \text{length}([a_i, b_i] \cap [0, x])/x\)

\(\mu\)'s pushforward under \(P\) is

\[P_*(\mu)(B_T) = \int_{x \in [0.1, 1.1]} B_T \cap [0, x] * \mu(dx)\]
More on Markov Kernels

• We can use Markov kernels to define the meanings of statements.

• A term can be seen as a Markov kernel that links the input variables (can be a distribution) with the output distribution.
Summary

• To reason about properties and correctness of probabilistic programs, we need semantics

• To define semantics, we can
  • Decompose it into semantics of program structures
  • Link it with mathematical concepts