Denotational Semantics for Probabilistic Programs

Hongseok Yang

KAIST

1 Introduction

In this lecture note, we describe the denotational semantics of a first-order programming language with discrete random choices. We focus mostly on spelling out all the details, which we covered in our lectures but did not appear in lecture slides.

2 Syntax and Typing Rules of a First-order Programming Language with Discrete Random Choices

Types	$\tau ::= \texttt{bool}$	Boolean Type
	$\mid \texttt{rational}$	Rational Type
	$\mid \texttt{dist}[\texttt{bool}]$	Boolean-Distribution Type
	$\mid \texttt{dist}[\texttt{rational}]$	Rational-Distribution Type

Expressions	$e ::= c$	Constants
	$\mid x$	Variables
	$\mid (p\ e_1 \ \ldots\ e_n)$	Function Applications
	$\mid (\texttt{let}\ [x\ e_1] e_2)$	Let Expressions
	$\mid (\texttt{if}\ e_0\ e_1\ e_2)$	Conditional Expressions

Constants	$c ::= 1.2 \mid \ldots$	Rational Numbers
	$\mid \texttt{true} \mid \texttt{false}$	Booleans

Primitive Operations	$p ::= \texttt{and} \mid \texttt{or} \mid \ldots$	Boolean Op.
	$\mid + \mid * \mid \ldots$	Arithematic Op.
	$\mid \texttt{flip} \mid \texttt{poisson} \mid \ldots$	Distribution Constr.
	$\mid \texttt{sample}_{\texttt{bool}} \mid \texttt{sample}_{\texttt{rational}}$	Sampling Op.

Typing Contexts	$\Gamma ::= x_1 : \tau_1, \ldots, x_n : \tau_n$
Typed Expressions	$\Gamma \vdash e : \tau$

Rules for typed expressions (or typing judgments):

$$\frac{\text{TypeConst}(c) = \tau}{\Gamma \vdash c : \tau} \qquad \frac{\Gamma(x) = \tau}{\Gamma \vdash x : \tau}$$

$$\frac{\text{TypePrimOp}(p) = (\tau_1, \ldots, \tau_n) \to \tau \qquad \Gamma \vdash e_i : \tau_i \text{ for all } i}{\Gamma \vdash (p\ e_1\ \ldots\ e_n) : \tau}$$

$$\frac{\Gamma \vdash e_1 : \tau_1 \qquad \Gamma, x : \tau_1 \vdash e_2 : \tau}{\Gamma \vdash (\texttt{let}\ [x\ e_1]\ e_2) : \tau}$$

$$\frac{\Gamma \vdash e_0 : \texttt{bool} \qquad \Gamma \vdash e_1 : \tau \qquad \Gamma \vdash e_2 : \tau}{\Gamma \vdash (\texttt{if}\ e_0\ e_1\ e_2) : \tau}$$

where

$\text{TypeConst}(\texttt{true}) = \texttt{bool}$
$\text{TypeConst}(\texttt{false}) = \texttt{bool}$
$\text{TypeConst}(1.2) = \texttt{rational}$
$\text{TypeConst}(-2) = \texttt{rational}$
$\ldots$

$\text{TypePrimOp}(\texttt{and}) = (\texttt{bool}, \texttt{bool}) \to \texttt{bool}$
$\text{TypePrimOp}(\texttt{or}) = (\texttt{bool}, \texttt{bool}) \to \texttt{bool}$
$\text{TypePrimOp}(+) = (\texttt{rational}, \texttt{rational}) \to \texttt{rational}$
$\text{TypePrimOp}(*) = (\texttt{rational}, \texttt{rational}) \to \texttt{rational}$
$\text{TypePrimOp}(\texttt{flip}) = (\texttt{rational}) \to \texttt{dist}[\texttt{bool}]$
$\text{TypePrimOp}(\texttt{poisson}) = (\texttt{rational}) \to \texttt{dist}[\texttt{rational}]$
$\text{TypePrimOp}(\texttt{sample}_{\texttt{bool}}) = (\texttt{dist}[\texttt{bool}]) \to \texttt{bool}$
$\text{TypePrimOp}(\texttt{sample}_{\texttt{rational}}) = (\texttt{dist}[\texttt{rational}]) \to \texttt{rational}$
$\ldots$

3 Semantics of Expressions

Three notations: for all booleans b, and all $a \in A$ and $c \in C$,

$$[b] \in \{0, 1\}$$
$$[b] = \begin{cases} 1 \text{ if } b = \mathrm{tt} \\ 0 \text{ if } b = \mathrm{ff} \end{cases}$$

$$\delta_a \in \mathrm{DiscProb}(A)$$
$$\delta_a(a') = \begin{cases} 1 \text{ if } a = a' \\ 0 \text{ otherwise} \end{cases}$$

$$\mathrm{bind} : \mathrm{DiscProb}(A) \times (A \to \mathrm{DiscProb}(C)) \to \mathrm{DiscProb}(C)$$
$$\mathrm{bind}(p, f)(c) = \sum_{a \in A} (p(a) \times f(a)(c)) = \mathbb{E}_{p(a)}[f(a)(c)]$$

Instead of $\mathrm{bind}(p, f)$, we often write

$$p \text{ bind } f.$$

Also, we use the λ expression to define mathematical functions in our definition of semantics.

The semantics of expressions is defined by induciton on the size of typing derivations.

$$\begin{aligned}
[\![\Gamma \vdash c : \tau]\!]\eta(v) &= \delta_c(v) \\
[\![\Gamma \vdash x : \tau]\!]\eta(v) &= \delta_{\eta(x)}(v) \\
[\![\Gamma \vdash (p\ e_1\ \ldots\ e_n) : \tau]\!]\eta &= ([\![\Gamma \vdash e_1 : \tau_1]\!]\eta) \text{ bind } \lambda v_1. \\
&\quad ([\![\Gamma \vdash e_2 : \tau_2]\!]\eta) \text{ bind } \lambda v_2. \\
&\quad \ldots \\
&\quad ([\![\Gamma \vdash e_n : \tau_n]\!]\eta) \text{ bind } \lambda v_n. \\
&\quad \mathrm{mean}(p)(v_1, \ldots, v_n) \\
[\![\Gamma \vdash (\texttt{let}\ [x\ e_1]\ e_2) : \tau]\!]\eta &= ([\![\Gamma \vdash e_1 : \tau_1]\!]\eta) \text{ bind } \lambda v_1. \\
&\quad [\![\Gamma \vdash e_2 : \tau_2]\!](\eta[x \mapsto v_1]) \\
[\![\Gamma \vdash (\texttt{if}\ e_0\ e_1\ e_2) : \tau]\!]\eta &= [\![\Gamma \vdash e_0 : \texttt{bool}]\!]\eta \text{ bind } \lambda b. \\
&\quad [b] \cdot [\![\Gamma \vdash e_1 : \tau]\!]\eta + [\neg b] \cdot [\![\Gamma \vdash e_2 : \tau]\!]\eta
\end{aligned}$$

Here we assume the definition of mean, which we will explain now. For a primitive operation p such that $\mathrm{TypePrimOp}(p) = (\tau_1, \ldots, \tau_n) \to \tau$, the $\mathrm{mean}(p)$ is a function of the following type:

$$\mathrm{mean}(p) : [\![\tau_1]\!] \times \ldots \times [\![\tau_n]\!] \to \mathrm{DiscProb}([\![\tau]\!])$$

It is defined as follows:

$$\begin{aligned}
\text{mean}(\texttt{and})(b_1, b_2)(b) &= \delta_{b_1 \wedge b_2}(b) \\
\text{mean}(\texttt{or})(b_1, b_2)(b) &= \delta_{b_1 \vee b_2}(b) \\
\text{mean}(+)(r_1, r_2)(r) &= \delta_{r_1 + r_2}(r) \\
\text{mean}(*)(r_1, r_2)(r) &= \delta_{r_1 * r_2}(r) \\
\text{mean}(\texttt{flip})(r)(p) &= \begin{cases} \delta_{[\text{tt} \mapsto r; \text{ff} \mapsto (1-r)]}(p) \text{ if } r \in [0,1] \\ \delta_{[\text{tt} \mapsto 1; \text{ff} \mapsto 0]}(p) \qquad \text{if } r \notin [0,1] \end{cases} \\
\text{mean}(\texttt{poisson})(r)(p) &= \begin{cases} \delta_{\text{Poisson}(r)}(p) \text{ if } r > 0 \\ \delta_{\text{Poisson}(1)}(p) \text{ otherwise} \end{cases} \\
\text{mean}(\texttt{sample}_{\texttt{bool}})(p)(v) &= p(v) \\
\text{mean}(\texttt{sample}_{\texttt{rational}})(p)(v) &= p(v)
\end{aligned}$$

4 Proving Equations using Semantics

One important application of the denotational semantics is to justify compiler optimisation; the semantics can be used to show that programs before and after the optimisation have the same meaning. In this section, we prove two program equations using the semantics that we have developed so far. The equations are not super exciting, but I hope that they help students see the workings of the semantics.

The first equation is:

$$[\![\Gamma \vdash (\texttt{if true } e_1 \ e_2) : t]\!] = [\![\Gamma \vdash e_1 : t]\!]. \tag{1}$$

To show this equality, pick an environment $\eta \in [\![\Gamma]\!]$. We will have to show that

$$[\![\Gamma \vdash (\texttt{if true } e_1 \ e_2) : t]\!]\eta = [\![\Gamma \vdash e_1 : t]\!]\eta. \tag{2}$$

To do so, we will use the following important property between the δ function and the bind operator:

$$\delta_a \text{ bind } f = f(a). \tag{3}$$

This can be easily proved by using the definitions of δ_a and bind. Using this property, we prove the equation (2):

$$\begin{aligned}
&[\![\Gamma \vdash (\texttt{if true } e_1 \ e_2) : t]\!]\eta \\
&\quad = ([\![\Gamma \vdash \texttt{true} : \texttt{bool}]\!]\eta) \text{ bind } (\lambda b.\, [b] \cdot [\![\Gamma \vdash e_1 : t]\!]\eta + [\neg b] \cdot [\![\Gamma \vdash e_2 : t]\!]\eta) \\
&\quad = \delta_{\text{tt}} \text{ bind } (\lambda b.\, [b] \cdot [\![\Gamma \vdash e_1 : t]\!]\eta + [\neg b] \cdot [\![\Gamma \vdash e_2 : t]\!]\eta) \\
&\quad = [\text{tt}] \cdot [\![\Gamma \vdash e_1 : t]\!]\eta + [\neg \text{tt}] \cdot [\![\Gamma \vdash e_2 : t]\!]\eta \qquad \text{by (3)} \\
&\quad = 1 \cdot [\![\Gamma \vdash e_1 : t]\!]\eta + 0 \cdot [\![\Gamma \vdash e_2 : t]\!]\eta \\
&\quad = [\![\Gamma \vdash e_1 : t]\!]\eta.
\end{aligned}$$

The second equation is:

$$\begin{aligned}&[\![\Gamma \vdash (\texttt{if}\ (\texttt{sample}\ (\texttt{flip}\ 0.2))\ \texttt{false}\ \texttt{true}) : \texttt{bool}]\!] \\ &\quad = [\![\Gamma \vdash (\texttt{sample}\ (\texttt{flip}\ 0.8)) : \texttt{bool}]\!]\end{aligned} \tag{4}$$

We first calculate the semantics of

$$\Gamma \vdash (\texttt{flip}\ 0.8) : \texttt{dist}[\texttt{bool}] \quad \text{and} \quad \Gamma \vdash (\texttt{sample}\ (\texttt{flip}\ 0.8)) : \texttt{bool}.$$

Here are the outcomes of these calculations:

$$\begin{aligned}&[\![\Gamma \vdash (\texttt{flip}\ 0.8) : \texttt{dist}[\texttt{bool}]]\!]\eta \\ &\quad = ([\![\Gamma \vdash 0.8 : \texttt{rational}]\!]\eta)\ \text{bind}\ (\lambda r.\, \text{mean}(\texttt{flip})(r)) \\ &\quad = ([\![\Gamma \vdash 0.8 : \texttt{rational}]\!]\eta)\ \text{bind}\ \left(\lambda r.\, \delta_{[\text{tt}\mapsto r;\text{ff}\mapsto 1-r]}\right) \\ &\quad = \delta_{0.8}\ \text{bind}\ \left(\lambda r.\, \delta_{[\text{tt}\mapsto r;\text{ff}\mapsto 1-r]}\right) \\ &\quad = \delta_{[\text{tt}\mapsto 0.8;\text{ff}\mapsto 0.2]} \qquad \text{by (3).}\end{aligned}$$

$$\begin{aligned}&[\![\Gamma \vdash (\texttt{sample}\ (\texttt{flip}\ 0.8)) : \texttt{bool}]\!]\eta \\ &= ([\![\Gamma \vdash (\texttt{flip}\ 0.8) : \texttt{dist}[\texttt{bool}]]\!]\eta)\ \text{bind}\ (\lambda d.\, \text{mean}(\texttt{sample})(d)) \\ &= \delta_{[\text{tt}\mapsto 0.8;\text{ff}\mapsto 0.2]}\ \text{bind}\ (\lambda d.\, \text{mean}(\texttt{sample})(d)) \qquad \text{by the first cal.} \\ &= \delta_{[\text{tt}\mapsto 0.8;\text{ff}\mapsto 0.2]}\ \text{bind}\ (\lambda d.\, d) \\ &= [\text{tt}\mapsto 0.8; \text{ff}\mapsto 0.2] \qquad \text{by (3).}\end{aligned}$$

By similar calculations, we can also show:

$$[\![\Gamma \vdash (\texttt{flip}\ 0.2) : \texttt{dist}[\texttt{bool}]]\!]\eta = [\text{tt}\mapsto 0.2; \text{ff}\mapsto 0.8]$$

Using what we have so far, we prove our target equation (4): for all $v \in [\![\texttt{bool}]\!]$,

$$\begin{aligned}&[\![\Gamma \vdash (\texttt{if}\ (\texttt{sample}\ (\texttt{flip}\ 0.2))\ \texttt{false}\ \texttt{true}) : \texttt{bool}]\!]\eta(v) \\ &\quad = \left(\begin{array}{l}([\![\Gamma \vdash (\texttt{sample}\ (\texttt{flip}\ 0.2)) : \texttt{bool}]\!]\eta)\ \text{bind}\ (\lambda b. \\ \quad [b] \cdot [\![\Gamma \vdash \texttt{false} : \texttt{bool}]\!]\eta + [\neg b] \cdot [\![\Gamma \vdash \texttt{true} : \texttt{bool}]\!]\eta)\end{array}\right)(v) \\ &\quad = \left(\begin{array}{l}[\text{tt}\mapsto 0.2; \text{ff}\mapsto 0.8]\ \text{bind}\ (\lambda b. \\ \quad [b] \cdot [\![\Gamma \vdash \texttt{false} : \texttt{bool}]\!]\eta + [\neg b] \cdot [\![\Gamma \vdash \texttt{true} : \texttt{bool}]\!]\eta)\end{array}\right)(v) \\ &\quad = \left(\begin{array}{l}0.2 \cdot ([\text{tt}] \cdot [\![\Gamma \vdash \texttt{false} : \texttt{bool}]\!]\eta(v) + [\neg \text{tt}] \cdot [\![\Gamma \vdash \texttt{true} : \texttt{bool}]\!]\eta(v)) \\ \quad + 0.8 \cdot ([\text{ff}] \cdot [\![\Gamma \vdash \texttt{false} : \texttt{bool}]\!]\eta(v) + [\neg \text{ff}] \cdot [\![\Gamma \vdash \texttt{true} : \texttt{bool}]\!]\eta(v))\end{array}\right) \\ &\quad = \left(\begin{array}{l}0.2 \cdot (1 \cdot [\![\Gamma \vdash \texttt{false} : \texttt{bool}]\!]\eta(v) + 0 \cdot [\![\Gamma \vdash \texttt{true} : \texttt{bool}]\!]\eta(v)) \\ \quad + 0.8 \cdot (0 \cdot [\![\Gamma \vdash \texttt{false} : \texttt{bool}]\!]\eta(v) + 1 \cdot [\![\Gamma \vdash \texttt{true} : \texttt{bool}]\!]\eta(v))\end{array}\right) \\ &\quad = 0.2 \cdot [\![\Gamma \vdash \texttt{false} : \texttt{bool}]\!]\eta(v) + 0.8 \cdot [\![\Gamma \vdash \texttt{true} : \texttt{bool}]\!]\eta(v) \\ &\quad = 0.2 \cdot \delta_{\text{ff}}(v) + 0.8 \cdot \delta_{\text{tt}}(v) \\ &\quad = [\text{tt}\mapsto 0.8; \text{ff}\mapsto 0.2](v) = [\![\Gamma \vdash (\texttt{sample}\ (\texttt{flip}\ 0.8)) : \texttt{bool}]\!]\eta(v).\end{aligned}$$