15-819: Foundations of Quantitative Program Analysis

Resource Analysis for Probabilistic Programs

Di Wang

November 24, 2019

1 Introduction

In previous lectures, we have studied automatic amortized resource analysis (AARA) for *deterministic* programs. In recent years, *probabilistic* programming becomes increasingly popular and has a wide range of applications such as randomized algorithms, cryptographic constructions, and machine learning. Reasoning rigorously about probabilistic programs introduces new possibilities and challenges. In terms of resource bound analysis, so far we have talked about *worst-case* resource usage, but when probabilities come into the picture, *average-case* resource usage makes more sense.

In this lecture, we extend AARA to derive *upper* bounds on the *expected* usage of monotone resources, such as time, for probabilistic programs. We extend our simple *functional* language that features pairs and lists with probabilistic constructs, adapting ideas from an automatic amortized resource analysis for *imperative* probabilistic programs [NCH18]. For simplicity, we only consider *linear* AARA, while it is straightforward to extend the type system with univariate and multivariate polynomial bounds. It will be clear that the extended type system has all the prominent features of AARA: precise, compositional, and efficient (through linear programming).

We also need a new cost semantics of probabilistic programs for formal development. We study two *operational* semantics introduced by Lago et al. [BLGS16]: one is *trace*-based, the other *distribution*-based. While the trace-based semantics is more intuitive as a direct extension of the evaluation dynamics with resource effects introduced in previous lectures, the distribution-based semantics is more suitable for presenting and proving the soundness theorem.

2 Probabilistic Programs

Syntax The syntax of our probabilistic programming language is given by the grammar below. It is a simple ML-like language with pairs and lists, except that an extra probabilistic-branching expression $\mathrm{flip}\{e_1; e_2\}(p)$ is introduced to execute e_1 or e_2 at random. The intuitive meaning of the flip expression is to flip a biased coin, which shows head with probability p, or tail with probability $(1-p)$, then execute e_1 if the coin shows head, or execute e_2 if the coin shows tail.

e	::=	x	x
		triv	$\langle\rangle$
		$\mathrm{fun}(f, x.e)$	$\mathsf{fun}\ f\,x = e$
		$\mathrm{app}(x_1; x_2)$	$x_1(x_2)$
		$\mathrm{pair}(x_1; x_2)$	$\langle x_1, x_2\rangle$
		$\mathrm{letp}(x; x_1, x_2.e)$	$\mathsf{letp}\,\langle x_1, x_2\rangle = x\ \mathsf{in}\ e$
		nil	$[\,]$
		$\mathrm{cons}(x_1; x_2)$	$x_1 :: x_2$
		$\mathrm{mat}_{\mathrm{L}}\{e_0; x_1, x_2.e_2\}(x)$	$\mathsf{case}\ x\ \{[\,] \hookrightarrow e_0 \mid x_1 :: x_2 \hookrightarrow e_2\}$
		$\mathrm{tick}\{q\}$	$\mathsf{tick}\ q$
		$\mathrm{let}(e_1; x.e_2)$	$\mathsf{let}\ x = e_1\ \mathsf{in}\ e_2$
		$\mathrm{share}(x; x_1, x_2.e)$	$\mathsf{share}\ x\ \mathsf{as}\ x_1, x_2\ \mathsf{in}\ e$
		$\mathrm{flip}\{e_1; e_2\}(p)$	$\mathsf{flip}\ p\ \{\mathsf{H} \hookrightarrow e_1 \mid \mathsf{T} \hookrightarrow e_2\}$

Elementary Probability Theory We recall some essential concepts from elementary probability theory. You can find more serious mathematical development of probabilities in textbooks on measure theory [Wil91, Bil12].

Consider a random experiment. Let Ω denote the set of all the possible outcomes, called the *sample space*. A *probability space* is a pair $(\Omega, \mathbb{P})$, where $\mathbb{P} : \Omega \to [0,1]$ is a *probability distribution* on Ω, i.e., $\sum_{\omega\in\Omega} \mathbb{P}(\omega) = 1$. The probability of an *event* $E \subseteq \Omega$, written $\mathbb{P}(E)$, is defined as $\sum_{\omega\in E} \mathbb{P}(\omega)$. We often write $\mathbb{P}(\theta)$ for the probability of a statement θ, i.e., $\mathbb{P}(\{\omega \mid \theta(\omega) \text{ is true}\})$. A *random variable* $X : \Omega \to \mathbb{R} \cup \{-\infty, +\infty\}$ is a function from a probability space to extended real numbers. The *expected value* of a random variable X is the weighted average $\mathbb{E}_{(\Omega,\mathbb{P})}(X) \coloneqq \sum_{\omega\in\Omega} X(\omega) \cdot \mathbb{P}(\omega)$. We often write $\mathbb{E}(X)$ if there is no ambiguity in the choice of the probability space. An important property of the expectation is *linearity*. If X and Y are random variables and $a, b \in \mathbb{R}$, then $(aX + bY)$ is a random variable and $\mathbb{E}(aX + bY) = a\mathbb{E}(X) + b\mathbb{E}(Y)$.

Example 1 (Experiments of Coin Flips)**.** *Consider an experiment that flips an unbiased coin twice. We can take* $\Omega = \{\mathsf{HH}, \mathsf{HT}, \mathsf{TH}, \mathsf{TT}\}$ *where* H *and* T *stands for the head and the tail of the coin, respectively. Because the coin is unbiased, the probability distribution is uniform over the sample space, i.e.,* $\mathbb{P}(\omega) = 1/4$ *for all* $\omega \in \Omega$*. As an instance of events and statements, we have*

$$\mathbb{P}(\textit{“at least one head is obtained”}) = \mathbb{P}(\{\mathsf{HH}, \mathsf{HT}, \mathsf{TH}\}) = \frac{3}{4}.$$

Consider another experiment that flips the same biased coin for n *times. Suppose the coin shows head with probability* p*, or tail with probability* $(1-p)$*. We take* $\Omega = \{\mathsf{H}, \mathsf{T}\}^n$*. For each coin flip sequence* $\omega = \omega_1 \cdots \omega_n \in \Omega$*, its probability is the product of the probability of the outcome of each coin flip, i.e.,* $\mathbb{P}(\omega) = p^{\sharp(i\le n : \omega_i = \mathsf{H})}(1-p)^{\sharp(i\le n : \omega_i = \mathsf{T})}$*, where* $\sharp(A)$ *returns the number of elements in the set* A*. As an instance of events and statements, we have*

$$\begin{aligned}\mathbb{P}(\textit{“exact } k \textit{ heads are obtained”}) &= \mathbb{P}(\{\omega \in \{\mathsf{H}, \mathsf{T}\}^n : \sharp(i \le n : \omega_i = \mathsf{H}) = k \wedge \sharp(i \le n : \omega_i = \mathsf{T}) = n - k\}) \\ &= \binom{n}{k} p^k (1-p)^{n-k}.\end{aligned}$$

Now we present several probabilistic programs and analyze their resource usage manually.

Example 2 (Biased Random Walk with One End)**.** *The following program implements a 1-dimensional random walk. We represent a natural number* $n \in \mathbb{N}$ *by a unit list* $\ell = [\langle\rangle, \cdots, \langle\rangle]$ *of size* n*. If the current position* n *is zero, then the program terminates. Otherwise, it proceeds to* $(n-1)$ *with probability* $3/4$*, or* $(n+1)$ *with probability* $1/4$*. We use ticks to count the number of steps before the random walk reaches zero.*

```
fun brdwalk l =
  match l with
  | [] → ()
  | _ :: xs →
    let _ = tick 1 in
    match flip(3/4) with
    | H → brdwalk(xs)
    | T → brdwalk(() :: () :: xs)
```

First let's analyze the worst-case *bound on the ticks. If the program always observes tails from coin flips, i.e., the random walk always increases the current position by one, the program will not terminate. Indeed, the worst-case bound on the ticks is infinity. However, this worst case rarely happens, and the program indeed terminates with probability one, called* almost-sure *(a.s.) termination [FFH15].*

Now let's reason about the average-case *bound on the ticks. We take the sample space* Ω *as all possible executions of the program. Let* $\mathbb{P}$ *maps these executions to their probabilities with respect to the coin flips. Then we can define a random variable* $T : \Omega \to \mathbb{Q}_{\ge 0} \cup \{\infty\}$ *that maps an execution to the number of ticks at the termination of the execution. If an execution will not terminate, then we take the limit along the infinite execution, which in this example, is indeed* ∞*. The* average *number of ticks is then defined as the expected value* $\mathbb{E}(T)$ *in the probability space* $(\Omega, \mathbb{P})$*.*

Let's try to use recurrence relations to analyze $\mathbb{E}(T)$. *Let* $R(n)$ *be the expectation* $\mathbb{E}(T)$ *when* ℓ *is initially a unit list of size* n. *Obviously,* $R(0) = 0$. *For* $n \geq 1$, *we know the program executes* tick{1}, *and then recurses on a list of size* $(n-1)$ *with probability* $3/4$, *or a list of size* $(n+1)$ *with probability* $1/4$. *Because* $R(n)$ *accounts for the* average *case,* $R(n)$ *should consist of a weighted average of* $R(n+1)$ *and* $R(n-1)$ *with respect to the branching probabilities as follows*[1]

$$R(n) = 1 + \frac{3}{4}R(n-1) + \frac{1}{4}R(n+1) \quad \textit{for} \quad n \geq 1.$$

To solve the recurrence relation using the techniques in previous lectures, we still need to figure out an initial value $R(1)$. *If we solve it anyway by treating* $R(1)$ *as a constant, we obtain the following solution*

$$R(n) = \frac{R(1)-2}{2} \cdot 3^n + \frac{2-R(1)}{2} + 2n.$$

Intuitively, the expected number of steps for one reaches zero from one should be the same as that for one reaches one from two. By the linearity of expectations, therefore, it should hold that $R(2) = R(1) + R(1)$. *With the extra information, we derive* $R(1) = 2$ *and thus* $R(n) = 2n$.

Example 3 (Unbiased Random Walk with Two Ends)**.** *The following program implements a 1-dimensional random walk with two ends. We represent the current position by a pair of "distances" from the left end and the right end. As in the last example, we use unit lists* ℓ *and* r *to represent natural numbers. Note that in this example, we consider unbiased coin flips. It is a standard result from probability theory that the random walk terminates with probability one.*

```
fun urdwalk (l, r) =
  match l with
  | [] → ()
  | _ :: xs →
    match r with
    | [] → ()
    | _ :: ys →
      let _ = tick 1 in
      match flip(1/2) with
      | H → urdwalk(xs, () :: () :: ys)
      | T → urdwalk(() :: () :: xs, ys)
```

Similar to the last example, let $\mathbb{E}(T)$ *denote the expected value of ticks. Let* $m = |\ell|$ *and* $n = |r|$ *be the sizes of the lists that indicate the current position. Let* $R(m,n)$ *be the expectation* $\mathbb{E}(T)$ *when* ℓ *and* r *are initially units lists of size* m *and* n, *respectively. Then we extract the following recurrence relation*

$$\begin{aligned} R(0,n) &= 0, \\ R(m,0) &= 0, \\ R(m,n) &= 1 + \frac{1}{2}R(m-1,n+1) + \frac{1}{2}R(m+1,n-1) \quad \textit{for} \quad n \geq 1, m \geq 1. \end{aligned}$$

We claim that $R(m,n) = m \cdot n$ *is a solution to the system above. Indeed, for the third equation, we have*

$$\begin{aligned} 1 + \frac{1}{2}R(m-1,n+1) + \frac{1}{2}R(m+1,n-1) &= 1 + \frac{1}{2}(m-1)(n+1) + \frac{1}{2}(m+1)(n-1) \\ &= 1 + \frac{1}{2}(m \cdot n + m - n - 1 + m \cdot n - m + n - 1) \\ &= 1 + m \cdot n - 1 \\ &= m \cdot n. \end{aligned}$$

[1] Indeed, this can be justified by the *law of total expectations*: $\mathbb{E}(X) = \mathbb{E}(X \mid A)\mathbb{P}(A) + \mathbb{E}(X \mid B)\mathbb{P}(B)$, where $\{A, B\}$ is a partition of Ω and $\mathbb{E}(X \mid A)$ is the *conditional* expectation of X on the subset A of the sample space.

Example 4 (Problems with Negative Ticks)**.** *The following program implements an unbiased 1-dimensional random walk with one termination position at zero. It is a known result from probability theory that the random walk terminates with probability one, however, the expected number of recursions (aka steps) before termination is infinity.*

```
fun nrdwalk l =
  match l with
  | [] → ()
  | _ :: xs →
    match flip(1/2) with
    | H → let _ = tick 1 in nrdwalk(xs)
    | T → let _ = tick -1 in nrdwalk(() :: () :: xs)
```

We no longer use ticks to count the number of steps before reaching zero. Instead, we use ticks to keep track of how many steps the walker has taken towards zero. *Intuitively, when the program terminates, the net cost should coincide with the size $|\ell|$ of the initial list ℓ.*

Because the program uses negative ticks, we define $R(n)$ to take values in the resource monoid where $n = |\ell|$ is the size of the initial list ℓ. Then we extract the following recurrence relation

$$\begin{aligned} R(0) &= (0,0), \\ R(n) &= \frac{1}{2}((1,0)\cdot R(n-1)) + \frac{1}{2}((0,1)\cdot R(n+1)). \end{aligned}$$

We claim that $R(n) = (n,0)$ is a solution to the system above. Indeed, for the second equation, we have

$$\begin{aligned} \frac{1}{2}((1,0)\cdot R(n-1)) + \frac{1}{2}((0,1)\cdot R(n+1)) &= \frac{1}{2}((1,0)\cdot(n-1,0)) + \frac{1}{2}((0,1)\cdot(n+1,0)) \\ &= \frac{1}{2}(n,0) + \frac{1}{2}(n,0) \\ &= (n,0). \end{aligned}$$

Let's consider a variant of the probabilistic program above.

```
fun nrdwalk' l =
  match l with
  | [] → ()
  | _ :: xs →
    let _ = tick -1 in
    match flip(1/2) with
    | H → let _ = tick 2 in nrdwalk'(xs)
    | T → nrdwalk'(() :: () :: xs)
```

Intuitively, when the variant terminates, the net cost should also be $|\ell|$ where ℓ is the initial list. However, we claim that $R(n) = (0,0)$ is an unsound *solution to the variant. Indeed, we have*

$$\begin{aligned} (0,1)\cdot(\frac{1}{2}((2,0)\cdot R(n-1)) + \frac{1}{2}R(n+1)) &= (0,1)\cdot(\frac{1}{2}((2,0)\cdot(0,0)) + \frac{1}{2}(0,0)) \\ &= (0,1)\cdot(1,0) \\ &= (0,0). \end{aligned}$$

*Although we can interchange the order of $\mathbb{E}(\cdot)$ and $+$ by linearity of expectations, we can*not *interchange the order of $\mathbb{E}(\cdot)$ and the resource monoid composition. As pointed out by Chatterjee et al. [WFG+19], negative ticks need to be treated differently in resource analysis for probabilistic programs via techniques from martingale theory. In this lecture, we will not cover this topic and simply assume all the ticks are nonnegative.*

3 Linear AARA with Probabilistic Branching

An Informal Account Let's try to analyze probabilistic programs with the potential method. Recall that $\Phi : \mathsf{State} \to \mathbb{Q}_{\geq 0}$ is a potential function if for all state S and operation o, the following

holds

$$\Phi(S) \geq cost(o) + \Phi(o(S)).$$

Now let's take probabilities into account. If operation o is probabilistic, then $cost(o)$ and $o(S)$ are also probabilistic. Intuitively, we want to extend the property as follows

$$\Phi(S) \geq \mathbb{E}(cost(o)) + \mathbb{E}(\Phi(o(S))),$$

where the expectation is taken with respect to the probability space induced by operation o from S. In addition, we assume that if S is a termination state, then $\Phi(S) = 0$. Then we have

$$\begin{aligned}
\mathbb{E}(\sum_i cost(o_i)) &= \sum_i \mathbb{E}(cost(o_i)) \\
&= \sum_i \mathbb{E}(\mathbb{E}(cost(o_i) \mid S_{i-1})) \\
&\leq \sum_i \mathbb{E}(\Phi(S_{i-1}) - \mathbb{E}(\Phi(S_i) \mid S_{i-1})) \\
&= \sum_i (\mathbb{E}(\Phi(S_{i-1})) - \mathbb{E}(\Phi(S_i))) \\
&= \mathbb{E}(\Phi(S_0)) = \Phi(S_0).
\end{aligned}$$

Note that if a probabilistic program has non-terminating executions, the first step of the derivation above requires that $cost(o) \geq 0$ for all the operations o.

Static Semantics Recall the definition of annotated types used in linear AARA.

$$\begin{array}{rcll}
\tau & ::= & \text{unit} & \mathbf{1} \\
 & & \text{arr}(A;B) & A \to B \\
 & & \text{prod}(\tau_1;\tau_2) & \tau_1 \times \tau_2 \\
 & & L(A) & L^q(\tau) \\
A, B & ::= & \text{pot}(\tau;q) & \langle \tau, q \rangle
\end{array}$$

The typing judgment stills has the form $\Gamma; q \vdash e : A$, the intuitive meaning of which is that the potential given by Γ and q is sufficient to cover the *expected* evaluation cost of e and the *expected* potential of the evaluation result with respect to A.

We keep all the syntax-directed and structural rules of the type system for linear AARA *unchanged*, and later we will prove their soundness in the probabilistic setting. Now we just need to add an extra rule for probabilistic branching. As we discussed in the examples, for the expression $\text{flip}\{e_1; e_2\}(p)$, if e_1 requires Φ_1 potential and e_2 requires Φ_2 potential, the evaluation of the flip expression should need a weighted average of Φ_1 and Φ_2, i.e., $p \cdot \Phi_1 + (1-p) \cdot \Phi_2$. Following is the rule L:FLIP for probabilistic branching.

$$\frac{\Gamma \curlyvee (p \times \Gamma_1, (1-p) \times \Gamma_2) \qquad q = p \cdot q_1 + (1-p) \cdot q_2 \qquad \Gamma_1; q_1 \vdash e_1 : A \qquad \Gamma_2; q_2 \vdash e_2 : A}{\Gamma; q \vdash \text{flip}\{e_1; e_2\}(p) : A} \text{ (L:FLIP)}$$

We utilize the sharing relation $\tau \curlyvee (\tau_1, \tau_2)$ to express the weighted average. First we extend the sharing relation to typing contexts.

$$\frac{}{\cdot \curlyvee (\cdot, \cdot)} \text{ (SH:ENV-1)} \qquad \frac{\Gamma \curlyvee (\Gamma_1, \Gamma_2) \qquad \tau \curlyvee (\tau_1, \tau_2)}{\Gamma, x : \tau \curlyvee (\Gamma_1, x : \tau_1, \Gamma_2, x : \tau_2)} \text{ (SH:ENV-2)}$$

Then we need to introduce a mechanism to perform the "weighting", or "scaling", on the potential. It can be defined syntactically as follows. Intuitively, $p \times \tau$ (resp., $p \times A$) produces a type with as much potential as that of the original type τ (resp., A) scaled by the factor p.

$$\begin{aligned}
p \times \text{unit} &= \text{unit} \\
p \times \text{arr}(A;B) &= \text{arr}(A;B) \\
p \times \text{prod}(\tau_1;\tau_2) &= \text{prod}(p \times \tau_1; p \times \tau_2) \\
p \times L(A) &= L(p \times A) \\
p \times \text{pot}(\tau;q) &= \text{pot}(p \times \tau; p \cdot q)
\end{aligned}$$

Also, we extend the scaling operation to typing contexts.

$$p \times (\cdot) = \cdot$$
$$p \times (\Gamma, x : \tau) = p \times \Gamma, x : (p \times \tau)$$

The following lemmas ensure that the premise $\Gamma \curlyvee (p \times \Gamma_1, (1-p) \times \Gamma_2)$ in the rule L:FLIP correctly captures the weighted average of the potential in Γ_1 and Γ_2.

Lemma 1. *1. If $V : \Gamma$ and $\Gamma \curlyvee (\Gamma_1, \Gamma_2)$, then $\Phi(V : \Gamma) = \Phi(V : \Gamma_1) + \Phi(V : \Gamma_2)$.*

2. If $v : \tau$ (resp., $v : A$), then $\Phi(v : p \times \tau) = p \cdot \Phi(v : \tau)$ (resp., $\Phi(v : p \times A) = p \cdot \Phi(v : A)$).

3. If $V : \Gamma$, then $\Phi(V : p \times \Gamma) = p \cdot \Phi(V : \Gamma)$.

Example 5 (Biased Random Walk with One End)**.** *We rewrite the biased random walk example in formal syntax as follows.*

$$\begin{array}{lll} \mathit{brdwalk} & \equiv & \mathsf{fun}\ f\,\ell = \\ & & \mathsf{case}\ \ell\ \{[] \hookrightarrow \langle\rangle \\ & & \qquad \_ :: x_2 \hookrightarrow \\ & & \qquad \mathsf{let}\ \_ = \mathsf{tick}\ 1\ \mathsf{in} \\ & & \qquad \mathsf{flip}\ {}^3\!/_4\ \{\mathsf{H} \hookrightarrow f(x_2) \\ & & \qquad\qquad \mathsf{T} \hookrightarrow f(\langle\rangle :: \langle\rangle :: x_2)\}\} \end{array}$$

We want to derive a type like $\langle L^2(\mathbf{1}), 0\rangle \to \langle \mathbf{1}, 0\rangle$ for brdwalk*. If the coin flip shows head, we do not need extra constant potential to evaluate the function call.*

$$\frac{}{f : \langle L^2(\mathbf{1}), 0\rangle \to \langle \mathbf{1}, 0\rangle, x_2 : L^2(\mathbf{1}); 0 \vdash f(x_2) : \langle \mathbf{1}, 0\rangle}\ (\text{L:APP})$$

Otherwise, if the coin flip shows tail, we need four units of extra constant potential to construct the argument list.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x_2 : L^2(\mathbf{1}); 0 \vdash x_2 : \langle L^2(\mathbf{1}), 0\rangle}\ (\text{L:VAR})}{x_2 : L^2(\mathbf{1}); 2 \vdash \langle\rangle :: x_2 : \langle L^2(\mathbf{1}), 0\rangle}\ (\text{L:CONS})}{x_2 : L^2(\mathbf{1}); 4 \vdash \langle\rangle :: \langle\rangle :: x_2 : \langle L^2(\mathbf{1}), 0\rangle}\ (\text{L:CONS})}{x_2 : L^2(\mathbf{1}); 4 \vdash f(\langle\rangle :: \langle\rangle :: x_2) : \langle \mathbf{1}, 0\rangle}\ (\text{L:APP})$$

It is obvious by the definition of potential scaling that

$$L^2(\mathbf{1}) \curlyvee (\frac{3}{4} \times L^2(\mathbf{1}), (1 - \frac{3}{4}) \times L^2(\mathbf{1})) \quad \textit{and} \quad 1 = \frac{3}{4} \cdot 0 + \frac{1}{4} \cdot 4.$$

Now we are able to apply the rule L:FLIP.

$$\dfrac{\begin{array}{c} f : \langle L^2(\mathbf{1}), 0\rangle \to \langle \mathbf{1}, 0\rangle, x_2 : L^2(\mathbf{1}); 0 \vdash f(x_2) : \langle \mathbf{1}, 0\rangle \\ f : \langle L^2(\mathbf{1}), 0\rangle \to \langle \mathbf{1}, 0\rangle, x_2 : L^2(\mathbf{1}); 4 \vdash f(\langle\rangle :: \langle\rangle :: x_2) : \langle \mathbf{1}, 0\rangle \end{array}}{f : \langle L^2(\mathbf{1}), 0\rangle \to \langle \mathbf{1}, 0\rangle, x_2 : L^2(\mathbf{1}); 1 \vdash \mathsf{flip}\ {}^3\!/_4\ \{\mathsf{H} \hookrightarrow f(x_2) \mid \mathsf{T} \hookrightarrow f(\langle\rangle :: \langle\rangle :: x_2)\} : \langle \mathbf{1}, 0\rangle}\ (\text{L:FLIP})$$

Example 6 (A Variant of the Biased Random Walk)**.** *Let's consider a variant of the function* brdwalk *that flips the coin first and then performs pattern match on the list ℓ.*

$$\begin{array}{lll} \mathit{brdwalk}' & \equiv & \mathsf{fun}\ f\,\ell = \\ & & \mathsf{flip}\ {}^3\!/_4\ \{\mathsf{H} \hookrightarrow \mathsf{case}\ \ell\ \{[] \hookrightarrow \langle\rangle \mid \_ :: x_2 \hookrightarrow \mathsf{let}\ \_ = \mathsf{tick}\ 1\ \mathsf{in}\ f(x_2)\} \\ & & \qquad \mathsf{T} \hookrightarrow \mathsf{case}\ \ell\ \{[] \hookrightarrow \langle\rangle \mid \_ :: x_2 \hookrightarrow \mathsf{let}\ \_ = \mathsf{tick}\ 1\ \mathsf{in}\ f(\langle\rangle :: \langle\rangle :: x_2)\}\} \end{array}$$

Intuitively, the function brdwalk' *should have the same expected resource usage as* brdwalk *does. However, we can*not *find a derivation for $\cdot; 0 \vdash \mathit{brdwalk}' : \langle\langle L^2(\mathbf{1}), 0\rangle \to \langle \mathbf{1}, 0\rangle, 0\rangle$. One reason is that*

in order to type the tail case of the flip expression, we need non-zero units of constant potential to cover the cost of constructing the list $\langle\rangle :: \langle\rangle :: x_2$*. However, there is no constant potential when we are typing the flip expression, thus we can*not *"compensate" the cost of this case by reducing the constant potential used in the head case.*

Nevertheless, we can derive a type like $\langle L^2(\mathbf{1}), 1\rangle \to \langle \mathbf{1}, 0\rangle$*. For the head case, we have*

$$\dfrac{\dfrac{\dfrac{}{f : \langle L^2(\mathbf{1}),1\rangle \to \langle \mathbf{1},0\rangle, x_2 : L^2(\mathbf{1}); 1 \vdash f(x_2) : \langle \mathbf{1},0\rangle}\,\text{(L:App)}}{f : \langle L^2(\mathbf{1}),1\rangle \to \langle \mathbf{1},0\rangle, x : L^2(\mathbf{1}); 2 \vdash \mathsf{let}\ \_ = \mathsf{tick}\ 1\ \mathsf{in}\ f(x_2) : \langle \mathbf{1},0\rangle}\,\text{(L:Let)}}{f : \langle L^2(\mathbf{1}),1\rangle \to \langle \mathbf{1},0\rangle, \ell : L^2(\mathbf{1}); 0 \vdash \mathsf{case}\ \ell\ \{[] \hookrightarrow \langle\rangle \mid \_ :: x_2 \hookrightarrow \mathsf{let}\ \_ = \mathsf{tick}\ 1\ \mathsf{in}\ f(x_2)\} : \langle \mathbf{1},0\rangle}\,\text{(L:MatL)}$$

For the tail case, we have

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x_2 : L^2(\mathbf{1}); 0 \vdash x_2 : \langle L^2(\mathbf{1}),0\rangle}\,\text{(L:Var)}}{x_2 : L^2(\mathbf{1}); 2 \vdash \langle\rangle :: x_2 : \langle L^2(\mathbf{1}),0\rangle}\,\text{(L:Cons)}}{x_2 : L^2(\mathbf{1}); 4 \vdash \langle\rangle :: \langle\rangle :: x_2 : \langle L^2(\mathbf{1}),0\rangle}\,\text{(L:Cons)}}{x_2 : L^2(\mathbf{1}); 5 \vdash \langle\rangle :: \langle\rangle :: x_2 : \langle L^2(\mathbf{1}),1\rangle}\,\text{(L:Relax)}}{f : \langle L^2(\mathbf{1}),1\rangle \to \langle \mathbf{1},0\rangle, x_2 : L^2(\mathbf{1}); 5 \vdash f(\langle\rangle :: \langle\rangle :: x_2) : \langle \mathbf{1},0\rangle}\,\text{(L:App)}}{f : \langle L^2(\mathbf{1}),1\rangle \to \langle \mathbf{1},0\rangle, x_2 : L^2(\mathbf{1}); 6 \vdash \mathsf{let}\ \_ = \mathsf{tick}\ 1\ \mathsf{in}\ f(\langle\rangle :: \langle\rangle :: x_2) : \langle \mathbf{1},0\rangle}\,\text{(L:Let)}}{f : \langle L^2(\mathbf{1}),1\rangle \to \langle \mathbf{1},0\rangle, \ell : L^2(\mathbf{1}); 4 \vdash \mathsf{case}\ \ell\ \{[] \hookrightarrow \langle\rangle \mid \_ :: x_2 \hookrightarrow \mathsf{let}\ \_ = \mathsf{tick}\ 1\ \mathsf{in}\ f(\langle\rangle :: \langle\rangle :: x_2)\} : \langle \mathbf{1},0\rangle}\,\text{(L:MatL)}$$

Then we can conclude by the rule L:Flip *in the same way as the previous example.*

4 Dynamic Semantics

To define the expected resource usage of probabilistic programs, we formulate a cost semantics based on an evaluation dynamics. As before, our expressions are in share-let-normal form, so we use an evaluation environment $V : \mathsf{Var} \to \mathsf{Val}$. Also, since we only consider nonnegative ticks, we adapt from the cost dynamic lecture a dynamics that uses *resource effects*. Recall that the evaluation judgment has the form

$$V \vdash e \Downarrow v \mid q,$$

which intuitively means that the expression e evaluates to a value v with net cost q.

A First Attempt To extend the evaluation dynamics with probabilistic constructs, our first idea is to add a component that states the probability of *an* execution. We might write $V \vdash e \Downarrow^p v \mid q$ which means that there exists *an* execution where the expression e evaluates to a value v with net cost q and probability p. Below are two rules for probabilistic branching.

$$\dfrac{V \vdash e_1 \Downarrow^{p_1} v_1 \mid q_1}{V \vdash \mathsf{flip}\{e_1; e_2\}(p) \Downarrow^{p \cdot p_1} v_1 \mid q}\,\text{(E:Flip-1)} \qquad \dfrac{V \vdash e_2 \Downarrow^{p_2} v_2 \mid q_2}{V \vdash \mathsf{flip}\{e_1; e_2\}(p) \Downarrow^{(1-p)\cdot p_2} v_2 \mid q}\,\text{(E:Flip-2)}$$

However, the attempt does not work well because there might be multiple different executions with the same evaluation result, cost, and probability. For example, consider the following program

$$e \equiv \mathsf{flip}\ \frac{1}{2}\ \{\mathsf{H} \hookrightarrow \mathsf{tick}\ 2 \mid \mathsf{T} \hookrightarrow \mathsf{let}\ \_ = \mathsf{tick}\ 1\ \mathsf{in}\ \mathsf{tick}\ 1\}.$$

Although the program has two possible executions, there is only one valid evaluation relation under the dynamics, which is

$$\cdot \vdash e \Downarrow^{1/2} \langle\rangle \mid 2.$$

If we do not know the number of possible executions with the same evaluation relation, it will be hard to reason about the weighted average cost over all the possible executions.

Trace-Based Dynamics We solve this issue by an observation that an execution is uniquely determined by the *trace* of outcomes of the coin flips in the execution. Therefore, we further augment the evaluation relation with a component for traces, i.e., a finite sequence of elements of $\{\mathsf{H},\mathsf{T}\}$. The trace-based evaluation judgment then has the form

$$V;\sigma \vdash e \Downarrow^{p} v \mid q,$$

where V, e, σ can be seen as "inputs" and v, q, p are "outputs". The intuitive meaning is that under the environment V, with a sequence σ of coin-flip outcomes, the expression e evaluates to a value v with probability p.

Figure 1 presents the rules for this trace-based evaluation dynamics. We write [] for empty traces, $\sigma_1 @ \sigma_2$ for trace concatenation, and $\mathsf{H} :: \sigma$ or $\mathsf{T} :: \sigma$ to observe a new coin flip and prepend the outcome to σ. In the rule E:LET, we multiply the probabilities of an execution of e_1 and an execution of e_2, as well as concatenate their traces of coin flips.

Recall the example that makes our first attempt fail:

$$e \equiv \mathsf{flip}\ \frac{1}{2}\ \{\mathsf{H} \hookrightarrow \mathsf{tick}\ 2 \mid \mathsf{T} \hookrightarrow \mathsf{let}\ \_ = \mathsf{tick}\ 1\ \mathsf{in}\ \mathsf{tick}\ 1\}$$

Now with the trace-based dynamics, we can distinguish the two possible executions:

$$\frac{\cdot;[] \vdash \mathsf{tick}\ 2 \Downarrow^{1} \langle\rangle \mid 2}{\cdot;[\mathsf{H}] \vdash e \Downarrow^{1/2} \langle\rangle \mid 2}\ (\text{E:FLIP-1}) \qquad \frac{\cdot;[] \vdash \mathsf{let}\ \_ = \mathsf{tick}\ 1\ \mathsf{in}\ \mathsf{tick}\ 1 \Downarrow^{1} \langle\rangle \mid 2}{\cdot;[\mathsf{T}] \vdash e \Downarrow^{1/2} \langle\rangle \mid 2}\ (\text{E:FLIP-2})$$

The type soundness theorem is given below. It tells you nothing more than the original type soundness theorem for deterministic programs does. It makes a statement that for any terminating evaluation, the obtained value is consistent with the typing judgment.

Theorem 1 (Type Soundness)**.** *Let* $\Gamma; q \vdash e : A$ *and* $V : \Gamma$*. For all* v_0, p_0, q_0, σ_0 *satisfying* $V;\sigma_0 \vdash e \Downarrow^{p_0} v_0 \mid q_0$*, it holds that* $v_0 : A$*.*

Proof. By induction on the evaluation judgement. □

Recall that in order to reason about expected resource usage, we need a notion of *probability distributions* over executions. With the trace-based dynamics, we can indeed capture all the terminating executions. Intuitively, the result value v, the net cost q, and the probability p, are determined uniquely by the environment V, the expressions e, and the trace of coin flips σ.

Lemma 2. *For all* V*,* e *and* σ*, there exists at most one combination of* v*,* q *and* p *such that* $V;\sigma \vdash e \Downarrow^{p} v \mid q$*.*

Therefore, for fixed V and e, the set of all finite traces induces a "distribution" over terminating executions. We can extract a "distribution" $[\![e]\!]^{V}_{\Downarrow}$ on values v and costs q as follows:

$$[\![e]\!]^{V}_{\Downarrow}(v,q) \coloneqq \sum_{\sigma} p_\sigma \quad \text{where } \sigma\text{'s are finite traces satisfying } V;\sigma \vdash e \Downarrow^{p_\sigma} v \mid q.$$

Note that if there are non-terminating executions with non-zero probabilities, the map defined above is a *partial* distribution in the sense that the probabilities do *not* sum up to one. In other words, the probability that e diverges under environment V is $(1 - \sum_{(v,q)} [\![e]\!]^{V}_{\Downarrow}(v,q))$.

5 Soundness

With the trace-based evaluation dynamics, we might state the soundness theorem for probabilistic programs as follows.

Theorem 2 (Soundness of AARA)**.** *Let* $\Gamma; q \vdash e : A$ *and* $V : \Gamma$*. Then*

$$\Phi(V : \Gamma) + q \geq \sum_{\sigma_0 : V;\sigma_0 \vdash e \Downarrow^{p_0} v_0 \mid q_0} p_0 \cdot (\Phi(v_0 : A) + q_0).$$

$V;\sigma \vdash e \Downarrow^p v \mid q$ "in environment V, with trace σ, expression e evaluates to value v with cost q and probability p"

$$\frac{}{V;[] \vdash x \Downarrow^1 V(x) \mid 0}\ \text{(E:Var)} \qquad \frac{}{V;[] \vdash \mathrm{triv} \Downarrow^1 \langle\rangle \mid 0}\ \text{(E:Triv)}$$

$$\frac{}{V;[] \vdash \mathrm{fun}(f, x.e) \Downarrow^1 \mathsf{clo}(V; f, x.e) \mid 0}\ \text{(E:Fun)}$$

$$\frac{V(x_1) = \mathsf{clo}(V'; f, x.e) \qquad V(x_2) = v_2 \qquad V', f \mapsto \mathsf{clo}(V'; f, x.e), x \mapsto v_2; \sigma \vdash e \Downarrow^p v \mid q}{V;\sigma \vdash \mathrm{app}(x_1; x_2) \Downarrow^p v \mid q}\ \text{(E:App)}$$

$$\frac{V(x_1) = v_1 \qquad V(x_2) = v_2}{V;[] \vdash \mathrm{pair}(x_1; x_2) \Downarrow^1 \langle v_1, v_2\rangle \mid 0}\ \text{(E:Pair)}$$

$$\frac{V(x) = \langle v_1, v_2\rangle \qquad V, x_1 \mapsto v_1, x_2 \mapsto v_2; \sigma \vdash e \Downarrow^p v \mid q}{V;\sigma \vdash \mathrm{letp}(x; x_1, x_2.e) \Downarrow^p v \mid q}\ \text{(E:LetP)} \qquad \frac{}{V;[] \vdash \mathrm{nil} \Downarrow^1 [] \mid 0}\ \text{(E:Nil)}$$

$$\frac{V(x_1) = v_1 \qquad V(x_2) = v_2}{V;[] \vdash \mathrm{cons}(x_1; x_2) \Downarrow^1 v_1 :: v_2 \mid 0}\ \text{(E:Cons)} \qquad \frac{V(x) = [] \qquad V;\sigma \vdash e_0 \Downarrow^p v \mid q}{V;\sigma \vdash \mathrm{mat_L}\{x; e_0, x_1.x_2\}(e_1) \Downarrow^p v \mid q}\ \text{(E:MatL-1)}$$

$$\frac{V(x) = v_1 :: v_2 \qquad V, x_1 \mapsto v_1, x_2 \mapsto v_2; \sigma \vdash e_1 \Downarrow^p v \mid q}{V;\sigma \vdash \mathrm{mat_L}\{x; e_0, x_1.x_2\}(e_1) \Downarrow^p v \mid q}\ \text{(E:MatL-2)}$$

$$\frac{}{V;[] \vdash \mathrm{tick}\{q\} \Downarrow^1 \langle\rangle \mid q}\ \text{(E:Tick)}$$

$$\frac{V;\sigma_1 \vdash e_1 \Downarrow^{p_1} v_1 \mid q_1 \qquad V, x \mapsto v_1; \sigma_2 \vdash e_2 \Downarrow^{p_2} v_2 \mid q_2}{V;\sigma_1 @ \sigma_2 \vdash \mathrm{let}(e_1; x.e_2) \Downarrow^{p_1 \cdot p_2} v_2 \mid q_1 + q_2}\ \text{(E:Let)}$$

$$\frac{V(x) = v \qquad V, x_1 \mapsto v, x_2 \mapsto v; \sigma \vdash e \Downarrow^p v' \mid q}{V;\sigma \vdash \mathrm{share}(x; x_1, x_2.e) \Downarrow^p v' \mid q}\ \text{(E:Share)}$$

$$\frac{V;\sigma \vdash e_1 \Downarrow^{p_1} v_1 \mid q_1}{V;\mathsf{H} :: \sigma \vdash \mathrm{flip}\{e_1; e_2\}(p) \Downarrow^{p \cdot p_1} v_1 \mid q_1}\ \text{(E:Flip-1)}$$

$$\frac{V;\sigma \vdash e_2 \Downarrow^{p_2} v_2 \mid q_2}{V;\mathsf{T} :: \sigma \vdash \mathrm{flip}\{e_1; e_2\}(p) \Downarrow^{(1-p) \cdot p_2} v_2 \mid q_2}\ \text{(E:Flip-2)}$$

Figure 1: Cost semantics with resource effects, trace-based.

However, it is unclear how to prove the theorem by induction on the evaluation judgment. The reason is that we now have to deal with a collection of evaluation judgments, instead of one. Intuitively, the trace-based evaluation dynamics talks about *individual* executions, while the goal of our resource analysis for probabilistic programs is to reason about *aggregated* information over all possible executions. Indeed, we can develop another evaluation dynamics that deals with *distributions* of executions more directly.

A First Attempt We might start with another judgment $V \vdash e \Rightarrow \mu$ with μ defined as a distribution over pairs (v, q), where v is the evaluation result and q is the net cost. Then we just need one rule for probabilistic branching.

$$\frac{V \vdash e_1 \Rightarrow \mu_1 \qquad V \vdash e_2 \Rightarrow \mu_2}{V \vdash \mathsf{flip}\{e_1; e_2\}(p) \Rightarrow p \cdot \mu_1 + (1-p) \cdot \mu_2} \text{ (DE:FLIP)}$$

We denote the weighted sum of two distributions μ_1 and μ_2 by $p \cdot \mu_1 + (1-p) \cdot \mu_2$, defined as $\lambda\omega. p \cdot \mu_1(\omega) + (1-p) \cdot \mu_2(\omega)$. For the leaf cases, such as unit values, we introduce the following rule

$$\frac{}{V \vdash \mathsf{triv} \Rightarrow \delta(\langle\rangle, 0)} \text{ (DE:TRIV)}$$

where $\delta(\omega)$ denotes the *point distribution* on ω, defined as $\lambda\omega'.[\omega = \omega']$.

However, the attempt does not work well for almost-sure termination. The issue is that the inductive definition of such a distribution dynamics will fail is there is a non-terminating execution. Consider the following program

$$f \equiv \mathsf{fun}\ f\_ = \mathsf{flip}\ \frac{1}{2}\ \{\mathsf{H} \hookrightarrow \langle\rangle \mid \mathsf{T} \hookrightarrow f(\langle\rangle)\}$$

and suppose that we want to evaluation $f(\langle\rangle)$. There does not exist a distribution μ that $V \vdash f(\langle\rangle) \Rightarrow \mu$, because if we try to apply the rules inductively, we will end up with a derivation tree with an infinite depth.

$$\dfrac{\dfrac{\dfrac{}{V \vdash \langle\rangle \Rightarrow \delta(\langle\rangle, 0)}\text{ (DE:TRIV)} \qquad \dfrac{\vdots}{V \vdash f(\langle\rangle) \Rightarrow ???}\text{ (DE:APP)}}{V \vdash \mathsf{flip}\ 1/2\ \{\mathsf{H} \hookrightarrow \langle\rangle \mid \mathsf{T} \hookrightarrow f(\langle\rangle)\} \Rightarrow ???}\text{ (DE:FLIP)}}{V \vdash f(\langle\rangle) \Rightarrow ???}\text{ (DE:APP)}$$

Distribution-Based Dynamics To cope with possible non-termination executions, we develop a partial-evaluation-like dynamics. Different from the one from previous lectures, we *do* care about the evaluation results. Also, to be able to construct a "complete" semantics from the "partial" ones, we index the evaluation judgments by their derivation depth. Formally, the distribution-based evaluation dynamics has the form

$$V \vdash e \Rightarrow^n \mu.$$

The intuitive meaning is that the expression e reduces to a partial distribution with an at-most-n derivation depth. We use *partial* distributions, whose probabilities sum up to a number possibly less than one, because there could be terminating executions with a derivation tree whose depth is more than n. Now for the unit values, we have the following rule

$$\frac{n > 0}{V \vdash \mathsf{triv} \Rightarrow^n \delta(\langle\rangle, 0)} \text{ (DE:TRIV)}$$

and for the flip expressions, we have

$$\frac{V \vdash e_1 \Rightarrow^n \mu_1 \qquad V \vdash e_2 \Rightarrow^n \mu_2}{V \vdash \mathsf{flip}\{e_1; e_2\}(p) \Rightarrow^{n+1} p \cdot \mu_1 + (1-p) \cdot \mu_2} \text{ (DE:FLIP)}$$

Figure 2 presents the rules for this distribution-based evaluation dynamics. In addition to the syntax-directed rules, we introduce a special base case where $n = 0$ and μ is set to a zero distribution $\mathbf{0} \coloneqq \lambda\omega.0$. The intuition is that a valid derivation should apply at least one rule. The rule DE:LET composes the distribution-based semantics for e_1 and e_2. Intuitively, if e_1 evaluates to v_1 with cost q_1 and probability p_1 under environment V, as well as e_2 evaluates to v_2 with cost q_2 and probability p_2 under environment $V, x \mapsto v_1$, then $\mathrm{let}(e_1; x.e_2)$ evaluates to v_2 with cost $(q_1 + q_2)$ and probability $(p_1 \cdot p_2)$.

We can now approximate the distribution over terminating executions using the depth-indexed distributions.

Lemma 3. *If $V \vdash e \Rightarrow^n \mu_1$, $V \vdash e \Rightarrow^m \mu_2$ and $n \le m$, then $\mu_1 \le \mu_2$ pointwise. As a consequence, we can define $[\![e]\!]^V_{\Rightarrow} \coloneqq \sup\{\mu_n : V \vdash e \Rightarrow^n \mu_n\} = \lim_{n\to\infty} \mu_n$ as the partial distribution of all possible terminating executions of a probabilistic program e under environment V.*

Recall the example that makes our first attempt fail:

$$f \equiv \mathsf{fun}\ f\_ = \mathsf{flip}\ \frac{1}{2}\ \{\mathsf{H} \hookrightarrow \langle\rangle \mid \mathsf{T} \hookrightarrow f(\langle\rangle)\}$$

With the distribution-based dynamics, we can derive the following judgments:

$$\begin{aligned}
&V \vdash f(\langle\rangle) \Rightarrow^0 \mathbf{0},\\
&V \vdash f(\langle\rangle) \Rightarrow^3 \frac{1}{2} \cdot \delta(\langle\rangle, 0),\\
&V \vdash f(\langle\rangle) \Rightarrow^5 \frac{1}{2} \cdot \delta(\langle\rangle, 0) + \frac{1}{4} \cdot \delta(\langle\rangle, 0),\\
&\cdots\\
&V \vdash f(\langle\rangle) \Rightarrow^{2k+1} \sum_{i=1}^{k} (\frac{1}{2})^i \cdot \delta(\langle\rangle, 0).
\end{aligned}$$

Therefore, let k approach infinity, we derive that $[\![f(\langle\rangle)]\!]^V_{\Rightarrow} = \delta(\langle\rangle, 0)$, i.e., the program terminates with probability one, as well as the evaluation result is always unit, and the net cost is always zero.

As a remark, the distribution-based dynamics is equivalent to the trace-based one.

Theorem 3. *Let V be an environment and e be an expression. Then $[\![e]\!]^V_{\Rightarrow} = [\![e]\!]^V_{\Downarrow}$.*

Soundness We now restate and prove the soundness theorem using the distribution-based dynamics.

Theorem 4 (Soundness of AARA)**.** *Let $\Gamma; q \vdash e : A$ and $V : \Gamma$. Then*

$$\Phi(V : \Gamma) + q \ge \sum_{(v_0, q_0)} [\![e]\!]^V_{\Rightarrow}(v_0, q_0) \cdot (\Phi(v_0 : A) + q_0).$$

Proof. It suffices to prove for every $n \in \mathbb{N}$, if $V \vdash e \Rightarrow^n \mu$, then

$$\Phi(V : \Gamma) + q \ge \sum_{(v_0, q_0)} \mu(v_0, q_0) \cdot (\Phi(v_0 : A) + q_0).$$

By induction on n with inversion on $V \vdash e \Rightarrow^n \mu$ then inner induction on $\Gamma; q \vdash e : A$.

- If $n = 0$, then $\mu = \mathbf{0}$. Straightforward.
- Suppose the lemma holds for some $n \in \mathbb{N}$. Now we consider the case for $n + 1$. Below are the proofs for L:LET and L:FLIP.

$\boxed{V \vdash e \Rightarrow^n \mu \qquad \text{“in environment } V\text{, expression } e \text{ reduces to result distribution } \mu \text{ within } n \text{ steps}}$

$$\frac{}{V \vdash e \Rightarrow^0 \mathbf{0}}\ (\text{DE:BASE}) \qquad \frac{n > 0}{V \vdash x \Rightarrow^n \delta(V(x), 0)}\ (\text{DE:VAR}) \qquad \frac{n > 0}{V \vdash \text{triv} \Rightarrow^n \delta(\langle\rangle, 0)}\ (\text{DE:TRIV})$$

$$\frac{n > 0}{V \vdash \text{fun}(f, x.e) \Rightarrow^n \delta(\text{clo}(V; f, x.e), 0)}\ (\text{DE:FUN})$$

$$\frac{V(x_1) = \text{clo}(V'; f, x.e) \qquad V(x_2) = v_2 \qquad V', f \mapsto \text{clo}(V'; f, x.e), x \mapsto v_2 \vdash e \Rightarrow^n \mu}{V \vdash \text{app}(x_1; x_2) \Rightarrow^{n+1} \mu}\ (\text{DE:APP})$$

$$\frac{n > 0 \qquad V(x_1) = v_1 \qquad V(x_2) = v_2}{V \vdash \text{pair}(x_1; x_2) \Rightarrow^n \delta(\langle v_1, v_2\rangle, 0)}\ (\text{DE:PAIR})$$

$$\frac{V(x) = \langle v_1, v_2\rangle \qquad V, x_1 \mapsto v_1, x_2 \mapsto v_2 \vdash e \Rightarrow^n \mu}{V \vdash \text{letp}(x; x_1, x_2.e) \Rightarrow^{n+1} \mu}\ (\text{DE:LETP}) \qquad \frac{n > 0}{V \vdash \text{nil} \Rightarrow^n \delta([], 0)}\ (\text{DE:NIL})$$

$$\frac{n > 0 \qquad V(x_1) = v_1 \qquad V(x_2) = v_2}{V \vdash \text{cons}(x_1; x_2) \Rightarrow^n \delta(v_1 :: v_2, 0)}\ (\text{DE:CONS})$$

$$\frac{V(x) = [] \qquad V \vdash e_0 \Rightarrow^n \mu}{V \vdash \text{mat}_\text{L}\{x; e_0, x_1.x_2\}(e_1) \Rightarrow^{n+1} \mu}\ (\text{DE:MATL-1})$$

$$\frac{V(x) = v_1 :: v_2 \qquad V, x_1 \mapsto v_1, x_2 \mapsto v_2 \vdash e_1 \Rightarrow^n \mu}{V \vdash \text{mat}_\text{L}\{x; e_0, x_1.x_2\}(e_1) \Rightarrow^{n+1} \mu}\ (\text{DE:MATL-2})$$

$$\frac{n > 0}{V \vdash \text{tick}\{q\} \Rightarrow^n \delta(\langle\rangle, q)}\ (\text{DE:TICK})$$

$$\frac{V \vdash e_1 \Rightarrow^n \mu \qquad \forall (v_1, q_1) \in \text{supp}(\mu) \colon V, x \mapsto v_1 \vdash e_2 \Rightarrow^n \mu_{(v_1,q_1)}}{V \vdash \text{let}(e_1; x.e_2) \Rightarrow^{n+1} \sum_{(v_1,q_1)} \sum_{(v_2,q_2)} \mu(v_1, q_1) \cdot \mu_{(v_1,q_1)}(v_2, q_2) \cdot \delta(v_2, q_1 + q_2)}\ (\text{DE:LET})$$

$$\frac{V(x) = v \qquad V, x_1 \mapsto v, x_2 \mapsto v \vdash e \Rightarrow^n \mu}{V \vdash \text{share}(x; x_1, x_2.e) \Rightarrow^{n+1} \mu}\ (\text{DE:SHARE})$$

$$\frac{V \vdash e_1 \Rightarrow^n \mu_1 \qquad V \vdash e_2 \Rightarrow^n \mu_2}{V \vdash \text{flip}\{e_1; e_2\}(p) \Rightarrow^{n+1} p \cdot \mu_1 + (1 - p) \cdot \mu_2}\ (\text{DE:FLIP})$$

Figure 2: Cost semantics with resource effects, distribution-based.

– (L:Let) By assumption, we have $\Gamma_1; q \vdash e_1 : \mathrm{pot}(\tau; p)$, $\Gamma_2, x : \tau; p \vdash e_2 : A$, and $\Gamma = \Gamma_1, \Gamma_2$ for some Γ_1, Γ_2. By inversion, we have $V \vdash e_1 \Rightarrow^n \mu_1$, for all $(v_1, q_1) \in \mathrm{supp}(\mu_1)$, $V, x \mapsto v_1 \vdash e_2 \Rightarrow^n \mu_{(v_1,q_1)}$, and $\mu = \sum_{(v_1,q_1)} \sum_{(v_2,q_2)} \mu_1(v_1, q_1) \cdot \mu_{(v_1,q_1)}(v_2, q_2) \cdot \delta(v_2, q_1 + q_2)$. By induction hypothesis, we have

$$\begin{aligned}\Phi(V : \Gamma_1) + q &\geq \sum_{(v_1,q_1)} \mu_1(v_1, q_1) \cdot (\Phi(v_1 : \mathrm{pot}(\tau; p)) + q_1) \\ &= \sum_{(v_1,q_1)} \mu_1(v_1, q_1) \cdot (\Phi(v_1 : \tau) + q_1 + p),\end{aligned}$$

and also for all $(v_1, q_1) \in \mathrm{supp}(\mu_1)$,

$$\Phi(V, x \mapsto v_1 : \Gamma_2, x : \tau) + p \geq \sum_{(v_2,q_2)} \mu_{(v_1,q_1)}(v_2, q_2) \cdot (\Phi(v_2 : A) + q_2).$$

Thus

$$\begin{aligned}\Phi(V : \Gamma) + q &= \Phi(V : \Gamma_1) + q + \Phi(V : \Gamma_2) \\ &\geq \sum_{(v_1,q_1)} \mu_1(v_1, q_1) \cdot (\Phi(v_1 : \tau) + q_1 + p) + \Phi(V : \Gamma_2) \\ &\geq \sum_{(v_1,q_1)} \mu_1(v_1, q_1) \cdot (\Phi(v_1 : \tau) + q_1 + p + \Phi(V : \Gamma_2)) \\ &= \sum_{(v_1,q_1)} \mu_1(v_1, q_1) \cdot (q_1 + p + \Phi(V, x \mapsto v_1 : \Gamma_2, x : \tau)) \\ &\geq \sum_{(v_1,q_1)} \mu_1(v_1, q_1) \cdot (q_1 + \sum_{(v_2,q_2)} \mu_{(v_1,q_1)}(v_2, q_2) \cdot (\Phi(v_2 : A) + q_2)) \\ &\geq \sum_{(v_1,q_1)} \mu_1(v_1, q_1) \cdot (\sum_{(v_2,q_2)} \mu_{(v_1,q_1)}(v_2, q_2) \cdot (\Phi(v_2 : A) + q_2 + q_1)) \\ &= \sum_{(v_1,q_1)} \sum_{(v_2,q_2)} \mu_1(v_1, q_1) \cdot \mu_{(v_1,q_1)}(v_2, q_2) \cdot (\Phi(v_2 : A) + q_2 + q_1).\end{aligned}$$

On the other hand, we know

$$\begin{aligned}&\sum_{(v_0,q_0)} \mu(v_0, q_0) \cdot (\Phi(v_0 : A) + q_0) \\ &= \sum_{(v_0,q_0)} (\sum_{(v_1,q_1)} \sum_{(v_2,q_2)} \mu_1(v_1, q_1) \cdot \mu_{(v_1,q_1)}(v_2, q_2) \cdot \delta(v_2, q_1 + q_2))(v_0, q_0) \cdot (\Phi(v_0 : A) + q) \\ &= \sum_{(v_0,q_0)} (\sum_{(v_1,q_1)} \sum_{(v_2,q_2)} \mu_1(v_1, q_1) \cdot \mu_{(v_1,q_1)}(v_2, q_2) \cdot [v_0 = v_2 \wedge q_0 = q_1 + q_2]) \cdot (\Phi(v_0 : A) + q) \\ &= \sum_{(v_1,q_1)} \sum_{(v_2,q_2)} \mu_1(v_1, q_1) \cdot \mu_{(v_1,q_1)}(v_2, q_2) \cdot (\Phi(v_2 : A) + q_1 + q_2).\end{aligned}$$

Thus we conclude this case.

– (L:Flip) By assumption, we have $\Gamma \curlyvee (p \times \Gamma_1, (1-p) \times \Gamma_2)$, $q = p \cdot q_1 + (1-p) \cdot q_2$, $\Gamma_1; q_1 \vdash e_1 : A$, and $\Gamma_2; q_2 \vdash e_2 : A$. By inversion, we have $V \vdash e_1 \Rightarrow^n \mu_1$, $V \vdash e_2 \Rightarrow^n \mu_2$, and $\mu = p \cdot \mu_1 + (1-p) \cdot \mu_2$. By induction hypothesis, we have

$$\begin{aligned}\Phi(V : \Gamma_1) + q_1 &\geq \sum_{(v_0,q_0)} \mu_1(v_0, q_0) \cdot (\Phi(v_0 : A) + q_0), \\ \Phi(V : \Gamma_2) + q_2 &\geq \sum_{(v_0,q_0)} \mu_2(v_0, q_0) \cdot (\Phi(v_0 : A) + q_0).\end{aligned}$$

Thus we conclude this case by

$$\begin{aligned}\Phi(V : \Gamma) + q &= \Phi(V : p \times \Gamma_1) + \Phi(V : (1-p) \times \Gamma_2) + p \cdot q_1 + (1-p) \cdot q_2 \\ &= p \cdot \Phi(V : \Gamma_1) + (1-p) \cdot \Phi(V : \Gamma_2) + p \cdot q_1 + (1-p) \cdot q_2 \\ &\geq p \cdot (\sum_{(v_0,q_0)} \mu_1(v_0, q_0) \cdot (\Phi(v_0 : A) + q_0)) + (1-p) \cdot (\sum_{(v_0,q_0)} \mu_2(v_0, q_0) \cdot (\Phi(v_0 : A) + q_0)) \\ &= \sum_{(v_0,q_0)} (p \cdot \mu_1(v_0, q_0) + (1-p) \cdot \mu_2(v_0, q_0)) \cdot (\Phi(v_0 : A) + q_0) \\ &= \sum_{(v_0,q_0)} \mu(v_0, q_0) \cdot (\Phi(v_0 : A) + q_0).\end{aligned}$$

□

6 Non-Termination

So far we only consider terminating executions in the evaluation dynamics. Although we assume all the ticks are nonnegative, i.e., resource cannot become available during the evaluation, we hope to be able to show that the expected bounds on resources like time imply almost-sure termination.

Recall that in the lecture on partial evaluation, we introduce $\circ$ to represent some point during the evaluation. We can then enrich the distribution-based dynamics with partial evaluation by forcing the result distribution μ in the judgment $V \vdash e \Rightarrow^n \mu$ to be a *complete* distribution instead of a partial one. To achieve this, we extend μ to be distributions over $(\mathsf{Val} \cup \{\circ\}) \times (\mathbb{Q}_{\geq 0} \cup \{\infty\})$. Most of the rules stay unchanged, except the following two:

$$\frac{}{V \vdash e \Rightarrow^0 \delta(\circ, 0)} \text{ (PE:BASE)}$$

$$\frac{V \vdash e_1 \Rightarrow^n \mu \qquad \forall (v_1, q_1) \in \mathrm{supp}(\mu) \colon (v_1 \neq \circ) \implies V, x \mapsto v_1 \vdash e_2 \Rightarrow^n \mu_{(v_1,q_1)}}{V \vdash \mathrm{let}(e_1; x.e_2) \Rightarrow^{n+1} \sum_{q_1} \mu(\circ, q_1) \cdot \delta(\circ, q_1) + \sum_{(v_1,q_1):v_1 \neq \circ} \sum_{(v_2,q_2)} \mu(v_1, q_1) \cdot \mu_{(v_1,q_1)}(v_2, q_2) \cdot \delta(v_2, q_1 + q_2)} \text{ (PE:LET)}$$

However, it is unclear now how to define $[\![e]\!]^V_{\Rightarrow}$ by the limit of $\{\mu_n\}_{n \in \mathbb{N}}$ where $V \vdash e \Rightarrow^n \mu_n$ (recall that previously, if $n \leq m$ then $\mu_n \leq \mu_m$ pointwise). Apparently, we need to define a new ordering on complete distributions. We define $\mu_1 \sqsubseteq \mu_2$ as

$$(\forall v, q \colon (v \neq \circ) \implies \mu_1(v, q) \leq \mu_2(v, q)) \wedge (\forall q \colon \mu_1((\mathsf{Val} \cup \{\circ\}) \times [0, q]) \geq \mu_2((\mathsf{Val} \cup \{\circ\}) \times [0, q])).$$

Intuitively, for concrete values, the order above is the same as the pointwise order on partial distributions, but for divergence, we take the other direction—the property above implies that $\mu_1(\{\circ\} \times [0, q]) \geq \mu_2(\{\circ\} \times [0, q])$ for all $q \in \mathbb{Q}_{\geq 0} \cup \{\infty\}$. Since we assume nonnegative ticks, the probability that the cost is smaller than any q with respect to μ_1 should be greater than or equal to that with respect to μ_2.

Lemma 4. *1. The relation $\sqsubseteq$ defines a partial order on the distributions.*

2. Let $\{\mu_n\}_{n \in \mathbb{N}}$ be a sequence such that $\mu_1 \sqsubseteq \mu_2 \sqsubseteq \cdots \sqsubseteq \mu_n \sqsubseteq \cdots$. Then there exists a least distribution μ such that for all $n \in \mathbb{N}$, $\mu_n \sqsubseteq \mu$. Further, we denote μ by $\bigsqcup_{n \in \mathbb{N}} \mu_n$.

Proof. Let's consider the ω-chain completeness. Let $\nu(v, q) \coloneqq \lim_{n \to \infty} \mu_n(v, q)$ for all $v \neq \circ$ and q. Let $P \coloneqq \sum_{(v,q):v \neq \circ} \nu(v, q)$. We want to construct ν° to be the "limit" of $\{\lambda q.\mu_n(\circ, q)\}_{n \in \mathbb{N}}$. Define $f^\circ_n(q) \coloneqq \mu_n(\{\circ\} \times [0, q])$ for all n and q. Then for each $n \in \mathbb{N}$, f°_n is monotone and right-ω-continuous and for all $q \in \mathbb{Q}_{\geq 0} \cup \{\infty\}$, $\{f^\circ_n(q)\}_{n \in \mathbb{N}}$ is non-increasing. Let f° be the pointwise limit of $\{f^\circ_n\}_{n \in \mathbb{N}}$. Because both $\mathbb{Q}_{\geq 0} \cup \{\infty\}$ and $[0, 1]$ are ω-complete partially ordered sets, the right-ω-continuous functions between them also form an ω-complete partially ordered set. Therefore, f° is also right-ω-continuous, and we can define $\nu^\circ(q) \coloneqq f^\circ(q) - \lim_{q' \to q^-} f^\circ(q')$. The final step is to prove that $f^\circ(\infty) = 1 - P$. For each $n \in \mathbb{N}$, we have $f^\circ_n(\infty) = 1 - \sum_{(v,q):v \neq \circ} \mu_n(v, q)$. Thus $f^\circ(\infty) = \lim_{n \to \infty} f^\circ_n(\infty) = \lim_{n \to \infty}(1 - \sum_{(v,q):v \neq \circ} \mu_n(v, q)) = 1 - \lim_{n \to \infty} \sum_{(v,q):v \neq \circ} \mu_n(v, q) = 1 - \sum_{(v,q):v \neq \circ} \lim_{n \to \infty} \mu_n(v, q) = 1 - P$ by Monotone Convergence Theorem. □

Lemma 5. *If $V \vdash e \Rightarrow^n \mu_1$, $V \vdash e \Rightarrow^m \mu_2$ and $n \leq m$, then $\mu_1 \sqsubseteq \mu_2$ pointwise. As a consequence, we can define $[\![e]\!]^V_{\Rightarrow} \coloneqq \bigsqcup_{n \in \mathbb{N}} \mu_n$ as the distribution of all possible terminating and non-terminating executions of a probabilistic program e under environment V.*

Recall that in the soundness proof, we do an induction on the index n of $V \vdash e \Rightarrow^n \mu$. The reason why this approach works is that the expected cost with respect to μ is monotone and ω-continuous. Although it is unclear whether the continuity sill holds for $\sqsubseteq$ or not, we can prove the following weaker result that is sufficient for our soundness proof.

Lemma 6. *Let* $h(\mu) \coloneqq \sum_q \mu(\circ, q) \cdot q + \sum_{(v,q):v\neq\circ} \mu(v,q) \cdot (\Phi(v : A) + q)$. *Let* $\{\mu_n\}_{n\in\mathbb{N}}$ *be a sequence such that* $\mu_1 \sqsubseteq \mu_2 \sqsubseteq \cdots \sqsubseteq \mu_n \sqsubseteq \cdots$. *Let* $M \in \mathbb{R}_{\geq 0}$. *If* $h(\mu_n) \leq M$ *for all* $n \in \mathbb{N}$, *then* $h(\bigsqcup_{n\in\mathbb{N}} \mu_n) \leq M$.

Proof. Let $\mu \coloneqq \bigsqcup_{n\in\mathbb{N}} \mu_n$. Let's define $f_n^\circ(q) \coloneqq \mu_n((\mathsf{Val} \cup \{\circ\}) \times [0, q])$. Similar to the proof of ω-chain completeness, f_n° is monotone and right-ω-continuous for each $n \in \mathbb{N}$ and for all $q \in \mathbb{Q}_{\geq 0} \cup \{\infty\}$, $\{f_n^\circ(q)\}_{n\in\mathbb{N}}$ is non-increasing. Moreover, $f_n^\circ(\infty) = 1$ for all $n \in \mathbb{N}$. Now we extend the the domain of f_n° from $\mathbb{Q}_{\geq 0} \cup \{\infty\}$ to $\mathbb{R}_{\geq 0} \cup \{\infty\}$ as $g_n^\circ(r) = \lim_{q\to r^+} f_n^\circ(q)$. By the right-$\omega$-continuity, we know that $g_n^\circ(q) = f_n^\circ(q)$ for all $q \in \mathbb{Q}_{\geq 0} \cup \{\infty\}$. Therefore, $\sum_{(v,q)} \mu_n(v,q) \cdot q = \int (g_n^\circ(\infty) - g_n^\circ(r))\,dr$. Let g° be the pointwise limit of $\{g_n^\circ\}_{n\in\mathbb{N}}$, so g° is also right-ω-continuous. Thus

$$
\begin{aligned}
h(\mu) &= \sum_q \mu(\circ, q) \cdot q + \sum_{(v,q):v\neq\circ} \mu(v,q) \cdot (\Phi(v : A) + q) \\
&= \sum_{(v,q)} \mu(v,q) \cdot q + \sum_{(v,q):v\neq\circ} \mu(v,q) \cdot \Phi(v : A) \\
&= \int (g^\circ(\infty) - g^\circ(r))\,dr + \sum_{(v,q):v\neq\circ} \mu(v,q) \cdot \Phi(v : A) \\
&= \int (1 - g^\circ(r))\,dr + \sum_{(v,q):v\neq\circ} \mu(v,q) \cdot \Phi(v : A) \\
&= \int (1 - \lim_{n\to\infty} g_n^\circ(r))\,dr + \sum_{(v,q):v\neq\circ} \lim_{n\to\infty} \mu_n(v,q) \cdot \Phi(v : A) \\
&= \int \lim_{n\to\infty} (1 - g_n^\circ(r))\,dr + \sum_{(v,q):v\neq\circ} \lim_{n\to\infty} \mu_n(v,q) \cdot \Phi(v : A) \\
&= \lim_{n\to\infty} \int (1 - g_n^\circ(r))\,dr + \lim_{n\to\infty} \sum_{(v,q):v\neq\circ} \mu_n(v,q) \cdot \Phi(v : A).
\end{aligned}
$$

Since $h(\mu_n) = \int (g_n^\circ(\infty) - g_n^\circ(r))\,dr + \sum_{(v,q):v\neq\circ} \mu_n(v,q) \cdot \Phi(v : A) = \int (1 - g_n^\circ(r))\,dr + \sum_{(v,q):v\neq\circ} \mu_n(v,q) \cdot \Phi(v : A)$ and $h(\mu_n) \leq M$ for all $n \in \mathbb{N}$, we conclude that $h(\mu) \leq \sup_{n\in\mathbb{N}} h(\mu_n) \leq M$. □

Now we can strengthen the soundness theorem to capture both termination and non-termination.

Theorem 5 (Soundness of AARA)**.** *Let* $\Gamma; q \vdash e : A$ *and* $V : \Gamma$. *Then*

$$
\Phi(V : \Gamma) + q \geq \sum_{q_0} [\![e]\!]^V_{\Rightarrow}(\circ, q_0) \cdot q_0 + \sum_{(v_0,q_0):v_0\neq\circ} [\![e]\!]^V_{\Rightarrow}(v_0, q_0) \cdot (\Phi(v_0 : A) + q_0).
$$

Proof. By the lemma we just proved, it suffices to prove for every $n \in \mathbb{N}$, if $V \vdash e \Rightarrow^n \mu$, then

$$
\Phi(V : \Gamma) + q \geq \sum_{q_0} \mu(\circ, q_0) \cdot q_0 + \sum_{(v_0,q_0)} \mu(v_0, q_0) \cdot (\Phi(v_0 : A) + q_0).
$$

We can still proceed by induction on n with inversion on $V \vdash e \Rightarrow^n \mu$ then inner induction on $\Gamma; q \vdash e : A$. Below is the proof for L:Let.

- (L:Let) By assumption, we have $\Gamma_1; q \vdash e_1 : \mathrm{pot}(\tau; p)$, $\Gamma_2, x : \tau; p \vdash e_2 : A$, and $\Gamma = \Gamma_1, \Gamma_2$ for some Γ_1, Γ_2. By inversion, we have $V \vdash e_1 \Rightarrow^n \mu_1$, for all $(v_1, q_1) \in \mathrm{supp}(\mu_1)$ such that $v_1 \neq \circ$, $V, x \mapsto v_1 \vdash e_2 \Rightarrow^n \mu_{(v_1,q_1)}$, and $\mu = \sum_{q_1} \mu_1(\circ, q_1) \cdot \delta(\circ, q_1) + \sum_{(v_1,q_1):v_1\neq\circ} \sum_{(v_2,q_2)} \mu_1(v_1, q_1) \cdot \mu_{(v_1,q_1)}(v_2, q_2) \cdot \delta(v_2, q_1 + q_2)$. By induction hypothesis, we have

  $$
  \begin{aligned}
  \Phi(V : \Gamma_1) + q &\geq \sum_{q_1} \mu_1(\circ, q_1) \cdot q_1 + \sum_{(v_1,q_1):v_1\neq\circ} \mu_1(v_1, q_1) \cdot (\Phi(v_1 : \mathrm{pot}(\tau; p)) + q_1) \\
  &= \sum_{q_1} \mu_1(\circ, q_1) \cdot q_1 + \sum_{(v_1,q_1):v_1\neq\circ} \mu_1(v_1, q_1) \cdot (\Phi(v_1 : \tau) + q_1 + p),
  \end{aligned}
  $$

  and also for all $(v_1, q_1) \in \mathrm{supp}(\mu_1)$ such that $v_1 \neq \circ$,

  $$
  \Phi(V, x \mapsto v_1 : \Gamma_2, x : \tau) + p \geq \sum_{q_2} \mu_{(v_1,q_1)}(\circ, q_2) \cdot q_2 + \sum_{(v_2,q_2):v_2\neq\circ} \mu_{(v_1,q_1)}(v_2, q_2) \cdot (\Phi(v_2 : A) + q_2).
  $$

Thus

$$
\begin{aligned}
\Phi(V:\Gamma)+q &= \Phi(V:\Gamma_1)+q+\Phi(V:\Gamma_2)\\
&\geq \sum_{q_1}\mu_1(\circ,q_1)\cdot q_1+\sum_{(v_1,q_1):v_1\neq\circ}\mu_1(v_1,q_1)\cdot(\Phi(v_1:\tau)+q_1+p)+\Phi(V:\Gamma_2)\\
&\geq \sum_{q_1}\mu_1(\circ,q_1)\cdot q_1+\sum_{(v_1,q_1):v_1\neq\circ}\mu_1(v_1,q_1)\cdot(\Phi(v_1:\tau)+q_1+p+\Phi(V:\Gamma_2))\\
&= \sum_{q_1}\mu_1(\circ,q_1)\cdot q_1+\sum_{(v_1,q_1):v_1\neq\circ}\mu_1(v_1,q_1)\cdot(q_1+p+\Phi(V,x\mapsto v_1:\Gamma_2,x:\tau))\\
&\geq \sum_{q_1}\mu_1(\circ,q_1)\cdot q_1\\
&+\sum_{(v_1,q_1):v_1\neq\circ}\mu_1(v_1,q_1)\cdot(q_1+\sum_{q_2}\mu_{(v_1,q_1)}(\circ,q_2)\cdot q_2+\sum_{(v_2,q_2):v_2\neq\circ}\mu_{(v_1,q_1)}(v_2,q_2)\cdot(\Phi(v_2:A)+q_2))\\
&= \sum_{q_1}\mu_1(\circ,q_1)\cdot q_1\\
&+\sum_{(v_1,q_1):v_1\neq\circ}\mu_1(v_1,q_1)\cdot(\sum_{q_2}\mu_{(v_1,q_1)}(\circ,q_2)\cdot(q_1+q_2)+\sum_{(v_2,q_2):v_2\neq\circ}\mu_{(v_1,q_1)}(v_2,q_2)\cdot(\Phi(v_2:A)+q_1+q_2))\\
&= \sum_{q_1}\mu_1(\circ,q_1)\cdot q_1+\sum_{(v_1,q_1):v_1\neq\circ}\sum_{q_2}\mu_1(v_1,q_1)\cdot\mu_{(v_1,q_1)}(\circ,q_2)\cdot(q_1+q_2)\\
&+\sum_{(v_1,q_1):v_1\neq\circ}\sum_{(v_2,q_2):v_2\neq\circ}\mu_1(v_1,q_1)\cdot\mu_{(v_1,q_1)}(v_2,q_2)\cdot(\Phi(v_2:A)+q_1+q_2).
\end{aligned}
$$

On the other hand, we know

$$
\begin{aligned}
&\sum_{q_0}\mu(\circ,q_0)\cdot q_0+\sum_{(v_0,q_0):v_0\neq\circ}\mu(v_0,q_0)\cdot(\Phi(v_0:A)+q_0)\\
=&\sum_{q_0}(\sum_{q_1}\mu_1(\circ,q_1)\cdot\delta(\circ,q_1)+\sum_{(v_1,q_1):v_1\neq\circ}\sum_{(v_2,q_2)}\mu_1(v_1,q_1)\cdot\mu_{(v_1,q_1)}(v_2,q_2)\cdot\delta(v_2,q_1+q_2))(\circ,q_0)\cdot q_0\\
+&\sum_{(v_0,q_0):v_0\neq\circ}(\sum_{q_1}\mu_1(\circ,q_1)\cdot\delta(\circ,q_1)+\sum_{(v_1,q_1):v_1\neq\circ}\sum_{(v_2,q_2)}\mu_1(v_1,q_1)\cdot\mu_{(v_1,q_1)}(v_2,q_2)\cdot\delta(v_2,q_1+q_2))(v_0,q_0)\cdot(\Phi(v_0:A)+q_0)\\
=&\sum_{q_0}(\sum_{q_1}\mu_1(\circ,q_1)\cdot\delta(\circ,q_1)+\sum_{(v_1,q_1):v_1\neq\circ}\sum_{q_2}\mu_1(v_1,q_1)\cdot\mu_{(v_1,q_1)}(\circ,q_2)\cdot\delta(\circ,q_1+q_2))(\circ,q_0)\cdot q_0\\
+&\sum_{(v_0,q_0):v_0\neq\circ}(\sum_{(v_1,q_1):v_1\neq\circ}\sum_{(v_2,q_2):v_2\neq\circ}\mu_1(v_1,q_1)\cdot\mu_{(v_1,q_1)}(v_2,q_2)\cdot\delta(v_2,q_1+q_2))(v_0,q_0)\cdot(\Phi(v_0:A)+q_0)\\
=&(\sum_{q_1}\mu_1(\circ,q_1)\cdot q_1+\sum_{(v_1,q_1):v_1\neq\circ}\sum_{q_2}\mu_1(v_1,q_1)\cdot\mu_{(v_1,q_1)}(\circ,q_2)\cdot(q_1+q_2))\\
+&(\sum_{(v_1,q_1):v_1\neq\circ}\sum_{(v_2,q_2):v_2\neq\circ}\mu_1(v_1,q_1)\cdot\mu_{(v_1,q_1)}(v_2,q_2)\cdot(\Phi(v_2:A)+q_1+q_2)).
\end{aligned}
$$

Thus we conclude this case.

□

References

[Bil12] P. Billingsley. *Probability and Measure*. John Wiley & Sons, Inc., 2012.

[BLGS16] J. Borgström, U. D. Lago, A. D. Gordon, and M. Szymczak. A Lambda-Calculus Foundation for Universal Probabilistic Programming. In *Int. Conf. on Functional Programming (ICFP'16)*, 2016.

[FFH15] L. M. Ferrer Fioriti and H. Hermanns. Probabilistic Termination: Soundness, Completeness, and Compositionality. In *Princ. of Prog. Lang. (POPL'15)*, 2015.

[NCH18] V. C. Ngo, Q. Carbonneaux, and J. Hoffmann. Bounded Expectations: Resource Analysis for Probabilistic Programs. In *Prog. Lang. Design and Impl. (PLDI'18)*, 2018.

[WFG+19] P. Wang, H. Fu, A. K. Goharshady, K. Chatterjee, X. Qin, and W. Shi. Cost Analysis of Nondeterministic Probabilistic Programs. In *Prog. Lang. Design and Impl. (PLDI'19)*, 2019.

[Wil91] D. Williams. *Probability with Martingales*. Cambridge University Press, 1991.