Inferring Covariances for Probabilistic Programs*

Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Christoph Matheja

Software Modelling and Verification Group
RWTH Aachen University
{benjamin.kaminski,katoen,matheja}@cs.rwth-aachen.de

Abstract. We study weakest precondition reasoning about the (co)variance of outcomes and the variance of run–times of probabilistic programs with conditioning. For outcomes, we show that approximating (co)variances is computationally more difficult than approximating expected values. In particular, we prove that computing both lower and upper bounds for (co)variances is Σ_2^0–complete. As a consequence, neither lower nor upper bounds are computably enumerable. We therefore present invariant–based techniques that *do* enable enumeration of both upper and lower bounds, once appropriate invariants are found. Finally, we extend this approach to reasoning about run–time variances.

Keywords: probabilistic programs · covariance · run–time

1 Introduction

Probabilistic programs describe manipulations on uncertain data in a succinct way. They are normal–looking programs describing how to obtain a distribution over the outputs. Using mostly standard programming language constructs, a probabilistic program transforms a prior distribution into a posterior distribution. Probabilistic programs provide a structured means to describe e.g., Bayesian networks (from AI), random encryption (from security), or predator–prey models (from biology) [5] succinctly.

The posterior distribution of a program is mostly determined by approximate means such as Markov Chain Monte Carlo (MCMC) sampling using (variants of) the well–known Metropolis–Hasting approach. This yields estimates for various measures of interest, such as expected values, second moments, variances, covariances, and the like. Such estimates typically come with weak guarantees in the form of confidence intervals, asserting that with a certain confidence the measure has a certain value. In contrast to these weak guarantees, we aim at the *exact* inference of such measures and their bounds. We hereby focus both on correctness and on run–time analysis of probabilistic programs. Put shortly, we are interested in obtaining *quantitative* statements about the possible outcomes of programs well as their run times.

* This work was supported by the Excellence Initiative of the German federal and state government.

This paper studies reasoning about the (co)variance of outcomes and the variance of run–times of probabilistic programs. Our programs support sampling from discrete probability distributions, conditioning on the outcomes of experiments by observations [5], and unbounded while–loops[1]. In the first part of the paper, we study the *theoretical complexity* of obtaining (co)variances on outcomes. We show that obtaining bounds on (co)variances is computationally more difficult than for expected values. In particular, we prove that computing both upper *and lower* bounds for (co)variances of program outcomes is Σ_2^0–complete, thus *not recursively enumerable*. This contrasts the case for expected values where lower bounds *are recursively enumerable*, while only upper bounds are Σ_2^0–complete [7]. We also show that determining the precise values of (co)variances as well as checking whether the (co)variance is infinite are both Π_2^0–complete. These results rule out analysis techniques based on finite loop–unrollings as complete approaches for reasoning about the covariances of outcomes of probabilistic programs.

In the second part of the paper, we therefore develop a weakest precondition reasoning technique for obtaining covariances on outcomes and variances on run–times. As with deductive reasoning for ordinary sequential programs, the crux is to find suitable loop–invariants. We present a couple of invariant–based proof rules that provide a sound and complete method to computably enumerate both upper and lower bounds on covariances, once appropriate invariants are found. We establish similar results for variances of the run–time of programs. The results of this paper extend McIver and Morgans approach for obtaining expectations of probabilistic programs [10], recent techniques for expected run–time analysis [8], and complement results on termination analysis [7,4].

2 Preliminaries

We study approximating the covariance of two random variables (ranging over program states) after successful termination of a probabilistic program on a given input state. Our development builds upon the *conditional probabilistic guarded command language (cpGCL)* [6]—an extension of Dijkstra's guarded command language [3] endowed with probabilistic choice and conditioning constructs.

Definition 1 (cpGCL [6]). *Let* $\mathbb{V}$ *be a finite* ***set of program variables***[2]*. Then the* ***set of programs in cpGCL****, denoted* $\mathbb{P}$*, adheres to the grammar*

$$\mathbb{P} \;::=\; \texttt{skip} \;\big|\; \texttt{empty} \;\big|\; \texttt{diverge} \;\big|\; \texttt{halt} \;\big|\; x \coloneqq E \;\big|\; \mathbb{P}\texttt{;}\ \mathbb{P} \;\big|\; \texttt{if}\ (B)\ \{\mathbb{P}\}\ \texttt{else}\ \{\mathbb{P}\} \;\big|\; \{\mathbb{P}\}\ [p]\ \{\mathbb{P}\} \;\big|\; \texttt{while}\ (B)\ \{\mathbb{P}\} \;\big|\; \texttt{observe}\ B\ ,$$

where $x \in \mathbb{V}$*,* E *is an arithmetical expression over* $\mathbb{V}$*,* $p \in [0,\,1] \cap \mathbb{Q}$ *is a rational probability, and* B *is a Boolean expression over arithmetic expressions over* $\mathbb{V}$*.*

[1] This contrasts MCMC–based analysis, as this is restricted to bounded programs.

[2] We restrict ourselves to a finite set of program variables for reasons of cleanness of the presentation. In principle, a countable set of program variables could be allowed.

If a program C contains neither a probabilistic choice $\{C'\}$ $[p]$ $\{C''\}$ nor an `observe`*–statement, we say that C is **non–probabilistic**.*

We briefly go over the meaning of the language constructs. Furthermore, we assign each statement an execution time in order to reason about the *run–time* of programs. `skip` (`empty`) does nothing—i.e. does not alter the current variable valuations—and consumes one (no) unit of time. `diverge` is syntactic sugar for the certainly non–terminating program `while` (`true`) $\{$`skip`$\}$. `halt` consumes no unit of time and halts program execution immediately (even when encountered inside a loop). It represents an *improper* termination of the program. $x \coloneqq E$, C_1`;` C_2, `if` (B) $\{C_1\}$ `else` $\{C_2\}$, and `while` (B) $\{C'\}$ are standard variable assignment, sequential composition, conditional choice, and while–loop constructs. Assignments and guard evaluations consume one unit of time.

$\{C_1\}$ $[p]$ $\{C_2\}$ is a probabilistic choice construct: With probability p the program C_1 is executed and with probability $1 - p$ the program C_2 is executed. Flipping the p–coin itself consumes one unit of time. `observe` B is the conditioning construct. Whenever in the execution of a program, an `observe` B is encountered, such that the current variable valuation satisfies the guard B, nothing happens except that one unit of time is being consumed. If, however, an `observe` B is encountered along an execution trace that occurs with probability q, such that B is *not* satisfied, this trace is blocked as it is considered an *undesired execution*. The probabilities of the remaining execution traces are then conditioned to the fact that this undesired trace was not encountered, i.e. the probabilities of the remaining execution traces are renormalized by $1 - q$. We refer to encountering such an undesired execution as an *observation violation*. For more details on conditioning and its semantics, see [6].

Notice that we do not include non–deterministic choice constructs (as opposed to probabilistic choice construct) in our language, as we would then run into similar problems as in [6, Section 6] in the presence of conditioning.

Example 1 (Conditioning inside a Loop). Consider the following loop:

`while` $(c = 1)\{$ $\{c \coloneqq 0\}$ `[0.5]` $\{x \coloneqq x + 1\}$`;` `observe` $c = 1 \;\vee\; x \text{ is odd}$ $\}$

Without the `observe`–statement, this loop would generate a geometric distribution on x. By considering the `observe`–statement, this distribution is conditioned to the fact that after termination x is odd. △

Given a probabilistic program C, an initial state σ, and a random variable f mapping program states to positive reals, we could now ask: What is the *conditional* expected value of f after proper termination of program C on input σ, *given that no observation was violated during the execution*? An answer to this question is given by the conditional weakest pre–expectation calculus introduced in [6]. For summarizing this calculus, we first formally characterize the random variables f, commonly called *expectations* [10]:

Definition 2 (Expectations [10,6]). *Let $\mathbb{S} = \{\sigma \mid \sigma \colon \mathbb{V} \to \mathbb{Q}\}$, where $\mathbb{Q}$ is the set of rational numbers, be the **set of program states**.*[3] *Then the **set of***

[3] Notice that $\mathbb{S}$ is countable and computably enumerable as $\mathbb{V}$ is finite.

C	$\mathsf{wp}\,[C]\,(f)$	$\mathsf{rt}\,[C]\,(t)$
`skip`	f	$t[\tau/\tau+1]$
`empty`	f	t
`diverge`	$\mathbf{0}$	∞
`halt`	$\mathbf{0}$	$\mathbf{0}$
$x \coloneqq E$	$f[x/E]$	$t[x/E,\, \tau/\tau+1]$
C_1 `;` C_2	$\mathsf{wp}[C_1] \circ \mathsf{wp}\,[C_2]\,(f)$	$\mathsf{rt}[C_1] \circ \mathsf{rt}\,[C_2]\,(t)$
`if` (B) $\{C_1\}$ `else` $\{C_2\}$	$[B]\cdot\mathsf{wp}\,[C_1]\,(f)$ $+[\neg B]\cdot\mathsf{wp}\,[C_2]\,(f)$	$\big([B]\cdot\mathsf{rt}\,[C_1]\,(t)$ $+[\neg B]\cdot\mathsf{rt}\,[C_2]\,(t)\big)[\tau/\tau+1]$
$\{C_1\}\ [p]\ \{C_2\}$	$p\cdot\mathsf{wp}\,[C_1]\,(f)$ $+(1-p)\cdot\mathsf{wp}\,[C_2]\,(f)$	$\big(p\cdot\mathsf{rt}\,[C_1]\,(t)$ $+(1-p)\cdot\mathsf{rt}\,[C_2]\,(t)\big)[\tau/\tau+1]$
`while` (B) $\{C'\}$	$\mathsf{lfp}\,X.\ [\neg B]\cdot f$ $+[B]\cdot\mathsf{wp}\,[C']\,(X)$	$\mathsf{lfp}\,X.\ \big([\neg B]\cdot t$ $+[B]\cdot\mathsf{rt}\,[C']\,(X)\big)[\tau/\tau+1]$
`observe` B	$[B]\cdot f$	$[B]\cdot t[\tau/\tau+1]$

C	$\mathsf{wlp}\,[C]\,(f)$
`diverge`	$\mathbf{1}$
`halt`	$\mathbf{1}$
`while` (B) $\{C'\}$	$\mathsf{gfp}\,X.\ [\neg B]\cdot f + [B]\cdot\mathsf{wlp}\,[C']\,(X)$

Table 1. Definition of wp, wlp, and rt. $[x/E]$ is a syntactic replacement with $f[x/E]\,(\sigma) = f(\sigma[x \mapsto \sigma(E)])$. $[B]$ is the indicator function of B with $[B](\sigma) = 1$ if $\sigma \models B$, and $[B](\sigma) = 0$ otherwise. $F \circ H(f)$ is the functional composition of F and H applied to f. $\mathsf{lfp}\,X.\ F(X)$ ($\mathsf{gfp}\,X.\ F(X)$) is the least (greatest) fixed point of F with respect to $\preceq$. Definitions of wlp for the other language constructs are as for wp and thus omitted.

expectations *is defined as* $\mathbb{E} = \left\{f \;\middle|\; f\colon \mathbb{S} \to \mathbb{R}^{\infty}_{\geq 0}\right\}$*, and the* ***set of bounded expectations*** *is defined as* $\mathbb{E}_{\leq 1} = \{f \mid f\colon \mathbb{S} \to [0,\,1]\}$*. A* ***complete partial order*** $\preceq$ *on both* $\mathbb{E}$ *and* $\mathbb{E}_{\leq 1}$ *is given by* $f_1 \preceq f_2$ *iff* $\forall\, \sigma \in \mathbb{S}\colon f_1(\sigma) \leq f_2(\sigma)$*.*

The **weakest (liberal) pre–expectation transformer** $\mathsf{wp}\colon \mathbb{P} \to (\mathbb{E} \to \mathbb{E})$ ($\mathsf{wlp}\colon \mathbb{P} \to (\mathbb{E}_{\leq 1} \to \mathbb{E}_{\leq 1})$) is defined according to Table 1 (middle column). By means of these two transformers, we can give an answer to the question posed above: Namely, the fraction $\mathsf{wp}[C](f)(\sigma)/\mathsf{wlp}[C](\mathbf{1})(\sigma)$ is indeed the conditional expected value of f after termination of C on input σ, given that no observation was violated during C's execution [6]. Consequently, we define:

Definition 3 (Conditional Expected Values [6]). *Let* $C \in \mathbb{P}$*,* $\sigma \in \mathbb{S}$*, and* $f \in \mathbb{E}$*. Then the* ***conditional expected value*** *of* f *after executing* C *on input* σ *given that no observation was violated is defined as*[4]

$$\mathsf{E}_{[\![C]\!](\sigma)}\,(f) \;=\; \frac{\mathsf{wp}\,[C]\,(f)\,(\sigma)}{\mathsf{wlp}\,[C]\,(\mathbf{1})\,(\sigma)}\;.$$

[4] We make use of the convention that $\frac{0}{0} = 0$. Note that since our probabilistic choice is a discrete choice and our language does not support sampling from continuous distributions, the problematic case of "$\frac{0}{0}$" can only occur if executing C on input σ will result in a violation of an observation with probability 1.

Having the definition for conditional expected values readily available, we can now turn towards defining the conditional (co)variance of a (two) random variables. We simply translate the textbook definition to our setting:

Definition 4 (Conditional (Co)variances). *Let $C \in \mathbb{P}$, $\sigma \in \mathbb{S}$, and $f, g \in \mathbb{E}$. Then the **conditional covariance** of the two random variables f and g after executing C on input σ, given that no observation was violated is defined as*

$$\mathbf{Cov}_{[\![C]\!](\sigma)}\,(\boldsymbol{f},\,\boldsymbol{g}) \;=\; \mathsf{E}_{[\![C]\!](\sigma)}\,(f \cdot g) - \mathsf{E}_{[\![C]\!](\sigma)}\,(f) \cdot \mathsf{E}_{[\![C]\!](\sigma)}\,(g) \ .$$

*The **conditional variance** of the single random variable f after executing C on input σ, given that no observation was violated is defined as the conditional covariance of f with itself, i.e. $\mathbf{Var}_{[\![C]\!](\sigma)}\,(\boldsymbol{f}) = \mathsf{Cov}_{[\![C]\!](\sigma)}\,(f,\,f)$.*

3 Computational Hardness of Computing (Co)variances

In this section, we will investigate the computational hardness of computing upper and lower bounds for conditional (co)variances. The results will be stated in terms of levels in the arithmetical hierarchy—a concept we first briefly recall:

Definition 5 (The Arithmetical Hierarchy [9,11]). *For every $n \in \mathbb{N}$, the **class $\boldsymbol{\Sigma_n^0}$** is defined as $\Sigma_n^0 = \{\mathcal{A} \mid \mathcal{A} = \{x \mid \exists y_1\, \forall y_2\, \exists y_3 \cdots \exists/\forall y_n \colon\ (x,\, y_1,\, y_2,\, y_3,\, \ldots,\, y_n) \in \mathcal{R}\},\ \mathcal{R} \text{ is a decidable relation}\}$ and the **class $\boldsymbol{\Pi_n^0}$** is defined as $\Pi_n^0 = \{\mathcal{A} \mid \mathcal{A} = \{x \mid \forall y_1\, \exists y_2\, \forall y_3 \cdots \exists/\forall y_n \colon\ (x,\, y_1,\, y_2,\, y_3,\, \ldots,\, y_n) \in \mathcal{R}\},\ \mathcal{R} \text{ is a decidable relation}\}$. Note that we require the values of variables to be drawn from a computable domain.* Multiple consecutive quantifiers of the same type *can be contracted to* one *quantifier of that type, so the number n really refers to the number of necessary* quantifier alternations. *A set $\mathcal{A}$ is called **arithmetical**, iff $\mathcal{A} \in \Gamma_n^0$, for $\Gamma \in \{\Sigma,\, \Pi\}$ and $n \in \mathbb{N}$. The arithmetical sets form a strict hierarchy, i.e. $\Gamma_n^0 \subset \Gamma_{n+1}^0$ holds for $\Gamma \in \{\Sigma,\, \Pi\}$ and $n \geq 0$. Furthermore, note that $\Sigma_0^0 = \Pi_0^0$ is exactly the class of the decidable sets and Σ_1^0 is exactly the class of the computably enumerable sets.*

Next, we recall the concept of many–one reducibility and completeness:

Definition 6 (Many–One Reducibility and Completeness [11,13,2]). *Let $\mathcal{A}$, $\mathcal{B}$ be arithmetical sets and let X be some appropriate universe such that $\mathcal{A}, \mathcal{B} \subseteq X$. $\mathcal{A}$ is called **many–one reducible** (or simply **reducible**) to $\mathcal{B}$, denoted $\boldsymbol{\mathcal{A} \leq_{\mathrm{m}} \mathcal{B}}$, iff there exists a computable function $r \colon X \to X$, such that $\forall\, x \in X \colon \big(x \in \mathcal{A} \Longleftrightarrow r(x) \in \mathcal{B}\big)$. If r is a function such that r reduces $\mathcal{A}$ to $\mathcal{B}$, we denote this by $\boldsymbol{r \colon \mathcal{A} \leq_{\mathrm{m}} \mathcal{B}}$. Note that $\leq_{\mathrm{m}}$ is transitive.*

*$\mathcal{A}$ is called **$\boldsymbol{\Gamma_n^0}$–complete**, for $\Gamma \in \{\Sigma,\, \Pi\}$, iff both $\mathcal{A} \in \Gamma_n^0$ and $\mathcal{A}$ is **$\boldsymbol{\Gamma_n^0}$–hard**, meaning $\mathcal{C} \leq_{\mathrm{m}} \mathcal{A}$, for any set $\mathcal{C} \in \Gamma_n^0$. Note that if $\mathcal{B} \in \Gamma_n^0$ and $\mathcal{A} \leq_{\mathrm{m}} \mathcal{B}$, then $\mathcal{A} \in \Gamma_n^0$, too. Furthermore, note that if $\mathcal{A}$ is Γ_n^0–complete and $\mathcal{A} \leq_{\mathrm{m}} \mathcal{B}$, then $\mathcal{B}$ is necessarily Γ_n^0–hard. Lastly, note that if $\mathcal{A}$ is Σ_n^0–complete, then $\mathcal{A} \in \Sigma_n^0 \setminus \Pi_n^0$. Analogously, if $\mathcal{A}$ is Π_n^0–complete, then $\mathcal{A} \in \Pi_n^0 \setminus \Sigma_n^0$.*

In the following, we study the hardness of obtaining covariance approximations both from above and from below. Furthermore, we are interested in exact values of covariances as well as in deciding whether the covariance is infinite. In order to formally investigate the arithmetical complexity of these problems, we define four problem sets which relate to upper and lower bounds for covariances and to the question whether the covariance is infinite:

Definition 7 (Approximation Problems for Covariances). *We define the following decision problems:*

$$\begin{aligned}
(C,\, \sigma,\, f,\, g,\, q) \in \mathcal{LCOVAR} \quad &\Longleftrightarrow \quad \mathsf{Cov}_{[\![C]\!](\sigma)}\,(f,\, g) > q \\
(C,\, \sigma,\, f,\, g,\, q) \in \mathcal{RCOVAR} \quad &\Longleftrightarrow \quad \mathsf{Cov}_{[\![C]\!](\sigma)}\,(f,\, g) < q \\
(C,\, \sigma,\, f,\, g,\, q) \in \mathcal{COVAR} \quad &\Longleftrightarrow \quad \mathsf{Cov}_{[\![C]\!](\sigma)}\,(f,\, g) = q \\
(C,\, \sigma,\, f,\, g) \in {}^{\infty}\mathcal{COVAR} \quad &\Longleftrightarrow \quad \mathsf{Cov}_{[\![C]\!](\sigma)}\,(f,\, g) \in \{-\infty,\, +\infty\}
\end{aligned}$$

where $C \in \mathbb{P}$, $\sigma \in \mathbb{S}$, $f, g \in \mathbb{E}$, and $q \in \mathbb{Q}$.[5]

The first fact we establish about the hardness of computing upper and lower bounds of covariances is that this is at most Σ_2^0–hard, thus not harder than deciding whether a non–probabilistic program, i.e. a program without observations and probabilistic choice, does *not* terminate on all inputs, or deciding whether a probabilistic program terminates after an expected finite number of steps [12,7]. Formally, we establish the following results:

Lemma 1. *$\mathcal{LCOVAR}$ and $\mathcal{RCOVAR}$ are both in Σ_2^0.*

For proving Lemma 1, we revert to a fact established in [7]: All lower bounds for expected outcomes are computably enumerable. As a consequence, there exists a computable function $\mathsf{wp}^k\,[C]\,(f)\,(\sigma)$ that is ascending in k, such that for given $C \in \mathbb{P}$, $\sigma \in \mathbb{S}$, and $f \in \mathbb{E}$, we have

$$\begin{aligned}
\forall\, k \in \mathbb{N} \colon \mathsf{wp}^k\,[C]\,(f)\,(\sigma) \;&\leq\; \mathsf{wp}\,[C]\,(f)\,(\sigma), \quad \text{and} \\
\sup_{k \in \mathbb{N}} \mathsf{wp}^k\,[C]\,(f)\,(\sigma) \;&=\; \mathsf{wp}\,[C]\,(f)\,(\sigma)\;.
\end{aligned}$$

Intuitively, for every $k \in \mathbb{N}$ the function $\mathsf{wp}^k\,[C]\,(f)\,(\sigma)$ outputs a lower bound of $\mathsf{wp}\,[C]\,(f)\,(\sigma)$ in ascending order.

Similarly, lower bounds for $\mathsf{wlp}\,[C]\,(\mathbf{1})\,(\sigma)$ can be enumerated. To see this, note that $\mathsf{wp}\,[C]\,(\mathbf{1})\,(\sigma) = 1$ for any `observe`–free program C and any state σ. $\mathsf{wp}\,[C]\,(\mathbf{1})\,(\sigma)$ can only be decreased by violation of an observation. Informally,

$$\mathsf{wp}\,[C]\,(\mathbf{1})\,(\sigma) \;=\; 1 - \text{“Probability of } C \text{ violating an observation”}\;.$$

Lower bounds for the latter probability can be enumerated by successively exploring the computation tree of C on input σ and accumulating the probability mass of all execution traces that lead to a violation of an observation. As a

[5] Note that, for obvious reasons, we restrict to *computable* expectations f, g only.

consequence, there must exist a computable function $\mathsf{wlp}^k\,[C]\,(\mathbf{1})\,(\sigma)$ that is descending in k, such that for given $C \in \mathbb{P}$ and $\sigma \in \mathbb{S}$,

$$\forall\, k \in \mathbb{N}\colon \mathsf{wlp}\,[C]\,(\mathbf{1})\,(\sigma) \;\leq\; \mathsf{wlp}^k\,[C]\,(\mathbf{1})\,(\sigma), \quad \text{and}$$
$$\mathsf{wlp}\,[C]\,(\mathbf{1})\,(\sigma) \;=\; \inf_{k\in\mathbb{N}} \mathsf{wlp}^k\,[C]\,(\mathbf{1})\,(\sigma)\;.$$

Since $\mathsf{wp}^k\,[C]\,(f)\,(\sigma)$ is ascending and $\mathsf{wlp}^k\,[C]\,(\mathbf{1})\,(\sigma)$ is descending in k, the quotient ${}^{\mathsf{wp}^k[C](f)(\sigma)}/_{\mathsf{wlp}^k[C](\mathbf{1})(\sigma)}$ is ascending in k. We can now prove Lemma 1:

Proof (Lemma 1). For $\mathcal{LCOVAR} \in \Sigma_2^0$, consider $(C,\,\sigma,\,f,\,g,\,q) \in \mathcal{LCOVAR}$ iff

$$\exists\, k\; \forall\, \ell\colon\; \frac{\mathsf{wp}^k\,[C]\,(f\cdot g)\,(\sigma)}{\mathsf{wlp}^k\,[C]\,(\mathbf{1})\,(\sigma)} - \frac{\mathsf{wp}^\ell\,[C]\,(f)\,(\sigma)\cdot \mathsf{wp}^\ell\,[C]\,(g)\,(\sigma)}{\mathsf{wlp}^\ell\,[C]\,(\mathbf{1})\,(\sigma)^2} \;>\; q\;.$$

For the proof for $\mathcal{RCOVAR}$, see Appendix A.1. □

Regarding the hardness of deciding whether a given rational is equal to the covariance and the hardness of deciding non–finiteness of covariances, we establish that this is at most Π_2^0–hard, thus not harder than deciding whether a non–probabilistic program terminates on all inputs, or deciding whether a probabilistic program does *not* terminate after an expected finite number of steps [12,7]. Formally, we establish the following:

Lemma 2. *$\mathcal{COVAR}$ and ${}^{\infty}\mathcal{COVAR}$ are both in Π_2^0.*

Proof. See Appendix A.2. □

So far we provided upper bounds for the computational hardness of solving approximation problems for covariances. We now show that these bounds are tight in the sense that these problems are *complete* for their respective level of the arithmetical hierarchy. For that we need a Σ_2^0– and a Π_2^0–hard problem in order to perform the necessary reductions for proving the hardness results. Adequate problems are the problem of almost–sure termination and its complement:

Theorem 1 (Hardness of the Almost–Sure Termination Problem [7]). *Let $C \in \mathbb{P}$ be* `observe`*–free. Then C* ***terminates almost–surely*** *on input $\sigma \in \mathbb{S}$, iff it does so with probability 1. The* ***problem set $\mathcal{AST}$*** *is defined as $(C,\,\sigma) \in \mathcal{AST}$ iff C terminates almost–surely on input σ. We denote the* ***complement of $\mathcal{AST}$*** *by $\overline{\mathcal{AST}}$.*[6] *$\mathcal{AST}$ is Π_2^0–complete and $\overline{\mathcal{AST}}$ is Σ_2^0–complete.*

By reduction from $\overline{\mathcal{AST}}$ we now establish the following hardness results:

Lemma 3. *$\mathcal{LCOVAR}$ and $\mathcal{RCOVAR}$ are both Σ_2^0–hard.*

[6] Note that by "complement" we mean not exactly a set theoretic complement but rather all pairs $(C,\,\sigma)$ such that C does not terminate almost–surely on σ.

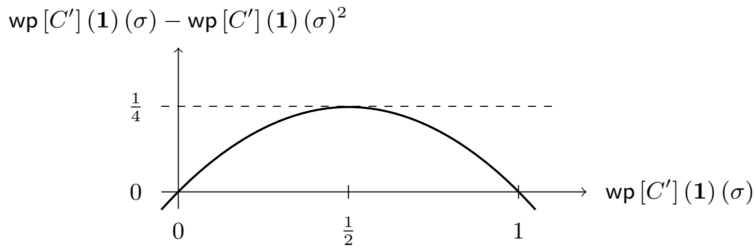


Fig. 1. Plot of the termination probability of a program against the resulting variance.

Proof. For proving the Σ_2^0–hardness of $\mathcal{LCOVAR}$, consider the reduction function $r_{\mathcal{L}}(C,\sigma) = (C', \sigma, v, v, 0)$, with $C' = v \coloneqq 0;\ \{\texttt{skip}\}\,[1/2]\,\{C\};\ v \coloneqq 1$, where variable v does not occur in C. Now consider the following:

$$\begin{aligned}
\mathsf{Cov}_{[\![C']\!](\sigma)}(v, v) &= \frac{\mathsf{wp}\,[C']\left(v^2\right)(\sigma)}{\mathsf{wlp}\,[C']\,(\mathbf{1})\,(\sigma)} - \frac{\mathsf{wp}\,[C']\,(v)\,(\sigma)^2}{\mathsf{wlp}\,[C']\,(\mathbf{1})\,(\sigma)^2} \\
&= \frac{\mathsf{wp}\,[C']\left(v^2\right)(\sigma)}{1} - \frac{\mathsf{wp}\,[C']\,(v)\,(\sigma)^2}{1^2} \qquad (C' \text{ is } \texttt{observe}\text{–free}) \\
&= \mathsf{wp}\,[C']\left(v^2\right)(\sigma) - \mathsf{wp}\,[C']\,(v)\,(\sigma)^2
\end{aligned}$$

Since v does not occur in C and v is set from 0 to 1 if and only if C' has terminated, this is equal to:

$$\begin{aligned}
&= \mathsf{wp}\,[C']\left(\mathbf{1}^2\right)(\sigma) - \mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma)^2 \\
&= \mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma) - \mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma)^2
\end{aligned}$$

Note that $\mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma)$ is exactly the probability of C' terminating on input σ. A plot of this termination probability against the resulting variance is given in Figure 1. We observe that $\mathsf{Cov}_{[\![C']\!](\sigma)}(v, v) = \mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma) - \mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma)^2 > 0$ iff C' terminates *neither* with probability 0 *nor* with probability 1. Since, however, C' terminates by construction *at least* with probability $1/2$, we obtain that $\mathsf{Cov}_{[\![C']\!](\sigma)}(v, v) > 0$ iff C' terminates with probability less than 1, which is the case iff C terminates with probability less than 1. Thus $r_{\mathcal{L}}(C,\sigma) = (C', \sigma, v, v, 0) \in \mathcal{LCOVAR}$ iff $(C,\sigma) \in \overline{\mathcal{AST}}$. Thus, $r_{\mathcal{L}} \colon \overline{\mathcal{AST}} \leq_{\mathrm{m}} \mathcal{LCOVAR}$. Since $\overline{\mathcal{AST}}$ is Σ_2^0–complete, if follows that $\mathcal{LCOVAR}$ is Σ_2^0–hard.

For the the proof for $\mathcal{RCOVAR}$, see Appendix A.3. □

A hardness results for $\mathcal{COVAR}$ is obtained by reduction from $\mathcal{AST}$.

Lemma 4. *$\mathcal{COVAR}$ is Π_2^0–hard.*

Proof. Similar to Lemma 3 using $r_{\mathcal{V}}(C,\sigma) = \left(C', \sigma, v, v, \frac{1}{4}\right)$, with $C' = v \coloneqq 0;$ $\{\texttt{diverge}\}\,[1/2]\,\{C\};\ v \coloneqq 1$. For details, see Appendix A.4. □

For a hardness result on $^{\infty}\mathcal{COVAR}$ we use the universal halting problem for non–probabilistic programs.

Theorem 2 (Hardness of the Universal Halting Problem [12]). *Let C be a non–probabilistic program. The* ***universal halting problem*** *is the problem of deciding whether C terminates on all inputs. Let* $\boldsymbol{\mathcal{UH}}$ *denote the* ***problem set****, defined as $C \in \mathcal{UH}$ iff $\forall \sigma \in \mathbb{S}$: C terminates on input σ. $\mathcal{UH}$ is Π_2^0–complete.*

We now establish by reduction from $\mathcal{UH}$ the remaining hardness result:

Lemma 5. *${}^\infty\mathcal{COVAR}$ is Π_2^0–hard.*

Proof. For proving the Π_2^0–hardness of ${}^\infty\mathcal{COVAR}$ we use the reduction function $r_\infty(C) = (C', \sigma, v, v)$, where σ is arbitrary but fixed and C' is the program

$c \coloneqq 1;\ i \coloneqq 0;\ x \coloneqq 0;\ v \coloneqq 0;\ \mathit{term} \coloneqq 0;\ \mathit{InitC};$
`while` $(c \neq$ `0`$)\{$
  $\mathit{StepC};$ `if` $(\mathit{term} = 1)\{\ v \coloneqq 2^x;\ i \coloneqq i+1;\ \mathit{term} \coloneqq 0;\ \mathit{InitC}\ \};$
  $\{c \coloneqq 0\}$ `[0.5]` $\{c \coloneqq 1\};\ x \coloneqq x+1\ \}$,

where *InitC* is a non–probabilistic program that initializes a simulation of the program C on input $e(i)$ (where $e\colon \mathbb{N} \to \mathbb{S}$ is some computable enumeration of $\mathbb{S}$), and *StepC* is a non–probabilistic program that does one single (further) step of that simulation and sets *term* to 1 if that step has led to termination of C.

Intuitively, the program C' starts by simulating C on input $e(0)$. During the simulation, it—figuratively speaking—gradually looses interest in further simulating C by tossing a coin after each simulation step to decide whether to continue the simulation or not. If eventually C' finds that C has terminated on input $e(0)$, it sets the variable v to a number exponential in the number of coin tosses that were made so far, namely to 2^x. C' then continues with the same procedure for the next input $e(1)$, and so on.

The variable x keeps track of the number of loop iterations (starting from 1 as the first loop iteration will definitely take place), which equals the number of coin tosses. The x–th loop iteration takes place with probability ${}^1/{2^x}$. The expected value $\mathsf{E}_{[\![C']\!](\sigma)}(v)$ is thus given by a series of the form $S = \sum_{i=1}^{\infty} {}^{v_i}/{2^i}$, where $v_i = 2^j$ for some $j \in \mathbb{N}$. Two cases arise:

(1) $C \in \mathcal{UH}$, i.e. C terminates on every input. In that case, v will infinitely often be updated to 2^x. Therefore, summands of the form ${}^{2^i}/{2^i}$ will appear infinitely often in S and so S diverges. Hence, the expected value of v is infinity and therefore, the variance of v must be infinite as well. Thus, $(C', \sigma, v, v) \in {}^\infty\mathcal{COVAR}$.

(2) $C \notin \mathcal{UH}$, i.e. there exists some input σ' with minimal $i \in \mathbb{N}$ such that $e(i) = \sigma'$ on which C does not terminate. In that case, the numerator of all summands of S is upper bounded by some constant 2^j and thus S converges. Boundedness of the v_i's implies that the series $\sum_{i=1}^{\infty} {}^{{v_i}^2}/{2^i} = \mathsf{E}_{[\![C']\!](\sigma)}\left(v^2\right)$ also converges. Hence, the variance of v is finite and $(C', \sigma, v, v) \notin {}^\infty\mathcal{COVAR}$. □

Lemmas 1 to 5 together directly yield the following completeness results:

Theorem 3 (The Hardness of Approximating Covariances).

1. *$\mathcal{LCOVAR}$ and $\mathcal{RCOVAR}$ are both Σ_2^0–complete.*
2. *$\mathcal{COVAR}$ and ${}^\infty\mathcal{COVAR}$ are both Π_2^0–complete.*

Remark 1 (The Hardness of Approximating Variances). It can be shown that *variance* approximation is not easier than covariance approximation: exactly the same completeness results as in Theorem 3 hold for analogous variance approximation problems. In fact, we have always reduced to approximating a variance for obtaining our hardness results on covariances. △

As an immediate consequence of Theorem 3, computing both upper and lower bounds for covariances is equally difficult. This is *contrary to the case for expected values*: While computing upper bounds for expected values is also Σ_2^0–complete, computing lower bounds is Σ_1^0–complete, thus lower bounds are computably enumerable [7]. Therefore we can computably enumerate an ascending sequence that converges to the sought–after expected value. By Theorem 3 this is *not possible* for a covariance as Σ_2^0–sets are in general not computably enumerable.

Theorem 3 rules out techniques based on finite loop–unrollings as *complete* approaches for reasoning about the covariances of outcomes of probabilistic programs. As this is a rather sobering insight, in the next section we will investigate invariant–aided techniques that are complete and can be applied to tackle these approximation problems.

4 Invariant–Aided Reasoning on Outcome Covariances

For straight–line (i.e. loop–free) programs, upper and lower bounds for covariances are obviously computable, e.g. by using the decompositions from Definitions 3 and 4, and the inference rules from Table 1. Problems do arise, however, for loops. We have seen in the previous section that neither upper nor lower bounds are computably enumerable. In this section we therefore present an invariant–aided approach for enumerating bounds on covariances of loops. The underlying principle of such techniques is quite commonly a result due to Park:

Theorem 4 (Park's Lemma [14]). *Let $(D, \sqsubseteq)$ be a complete partial order and $F\colon D \to D$ be continuous. Then, for all $d \in D$, it holds that $F(d) \sqsubseteq d$ implies $\mathsf{lfp}\, F \sqsubseteq d$, and $d \sqsubseteq F(d)$ implies $d \sqsubseteq \mathsf{gfp}\, F$.*

Using this theorem, we can verify in a relatively easy fashion that some element is an over–approximation of the least fixed point or an under–approximation of the greatest fixed point of a continuous mapping on a complete partial order. In the following, let $C = \texttt{while}\ (B)\ \{C'\}$. In order to exploit Park's Lemma for enumerating bounds on covariances for this while–loop, recall

$$
\begin{aligned}
\mathsf{Cov}_{\llbracket C \rrbracket(\sigma)}\,(f,\,g) \;&=\; \mathsf{E}_{\llbracket C \rrbracket(\sigma)}\,(f \cdot g) - \mathsf{E}_{\llbracket C \rrbracket(\sigma)}\,(f) \cdot \mathsf{E}_{\llbracket C \rrbracket(\sigma)}\,(g) \\
&=\; \frac{\mathsf{wp}\,[C]\,(f \cdot g)\,(\sigma)}{\mathsf{wlp}\,[C]\,(\mathbf{1})\,(\sigma)} - \frac{\mathsf{wp}\,[C]\,(f)\,(\sigma) \cdot \mathsf{wp}\,[C]\,(g)\,(\sigma)}{\mathsf{wlp}\,[C]\,(\mathbf{1})\,(\sigma)^2}\,.
\end{aligned}
$$

By inspection of the last line, we can see that for obtaining an over–approximation of $\mathsf{Cov}_{\llbracket C \rrbracket(\sigma)}\,(f,\,g)$, it suffices to over–approximate ${}^{\mathsf{wp}[C'](f \cdot g)(\sigma)}/_{\mathsf{wlp}[C'](\mathbf{1})(\sigma)}$, which can be done by over–approximating $\mathsf{wp}\,[C']\,(f \cdot g)\,(\sigma)$ and under–approximating $\mathsf{wlp}\,[C']\,(\mathbf{1})\,(\sigma)$. Since wp (wlp) of a loop is defined in terms of a least (greatest) fixed point, we can apply Park's Lemma for over–approximating this fraction. This leads us to the following proof rule:

Theorem 5 (Invariant–Aided Over–Approximation of Covariances). *Let* $C = \texttt{while}\ (B)\ \{C'\}$*,* $\sigma \in \mathbb{S}$*,* $f, g \in \mathbb{E}$*,* $F_h(X) = [\neg B] \cdot h + [B] \cdot \mathsf{wp}\,[C']\,(X)$*, where* $h \in \mathbb{E}$*, and* $G(Y) = [\neg B] + [B] \cdot \mathsf{wlp}\,[C']\,(Y)$*. Furthermore, let* $\widehat{X} \in \mathbb{E}$ *and* $\widehat{Y} \in \mathbb{E}_{\leq 1}$*, such that* $F_{f \cdot g}(\widehat{X}) \preceq \widehat{X}$*,* $\widehat{Y} \preceq G(\widehat{Y})$*, and* $\widehat{Y}(\sigma) > 0$*. Then for all* $k \in \mathbb{N}$ *it holds that*[7]

$$\mathsf{Cov}_{[\![C]\!](\sigma)}\,(f,\, g) \quad \leq \quad \frac{\widehat{X}(\sigma)}{\widehat{Y}(\sigma)} - \frac{F_f^k(\mathbf{0})(\sigma) \cdot F_g^k(\mathbf{0})(\sigma)}{G^k(\mathbf{1})(\sigma)^2}\;.$$

Proof. See Appendix A.5 □

By this method we can computably enumerate upper bounds for covariances once appropriate invariants are found. The catch is that if we choose the invariants, such that $F_{f \cdot g}(\widehat{X})(\sigma) < \widehat{X}(\sigma)$ or $\widehat{Y}(\sigma) < G(\widehat{Y})(\sigma)$, then the enumeration will *not* get arbitrarily close to the actual covariance. Note, however, that our method is complete since we could have chosen $\widehat{X} = \mathsf{lfp}\, F_{f \cdot g}$ and $\widehat{Y} = \mathsf{gfp}\, G$:

Corollary 1 (Completeness of Theorem 5). *Let* $C = \texttt{while}\ (B)\ \{C'\}$*,* $\sigma \in \mathbb{S}$*,* $f, g \in \mathbb{E}$*. Then there exist* $\widehat{X} \in \mathbb{E}$ *and* $\widehat{Y} \in \mathbb{E}_{\leq 1}$*, such that*

$$\inf_{k \in \mathbb{N}}\ \frac{\widehat{X}(\sigma)}{\widehat{Y}(\sigma)} - \frac{F_f^k(\mathbf{0})(\sigma) \cdot F_g^k(\mathbf{0})(\sigma)}{G^k(\mathbf{1})(\sigma)^2} \quad = \quad \mathsf{Cov}_{[\![C]\!](\sigma)}\,(f,\, g)\,.$$

By considerations analogous to the ones above, we can formulate dual results for lower bounds. For details, see Appendix A.6.

Example 2 (Application of Theorem 5). Reconsider the loop from Example 1. For reasoning about the variance of x, we pick the invariants

$$\begin{aligned}\widehat{X} \;=\; & [c \neq 0] \cdot x^2 + [c = 1] \cdot \Bigg([x \text{ is even}] \cdot \frac{1}{27}\left(9x^2 + 30x + 41\right) \\ & \qquad + [x \text{ is odd}] \cdot \frac{2}{27}\left(9x^2 + 12x + 20\right) \Bigg), \quad \text{and} \\ \widehat{Y} \;=\; & [c \neq 0] + [c = 1] \cdot \left([x \text{ is even}] \cdot \frac{1}{3} + [x \text{ is odd}] \cdot \frac{2}{3} \right)\;,\end{aligned}$$

which satisfy the preconditions of Theorem 5. If we enter the loop in a state σ with $\sigma(c) = 1$ and $\sigma(x) = 0$, we have $\widehat{X}(\sigma)/\widehat{Y}(\sigma) = 41/9$ which is our first upper bound. We can now enumerate further upper bounds by doing fixed point iteration on $F_x(X) = [c \neq 1] \cdot x + [c = 1] \cdot \mathsf{wp}\,[\textit{loop body}]\,(X) = [c \neq 1] \cdot x + [c = 1] \cdot \frac{1}{2}\big([x \text{ is odd}] \cdot X[c/0] + X[x/x+1]\big)$ and $G(Y) = [c \neq 1] + [c = 1] \cdot \mathsf{wlp}\,[\textit{loop body}]\,(Y) = [c \neq 1] + [c = 1] \cdot \frac{1}{2}\big([x \text{ is odd}] \cdot Y[c/0] + Y[x/x+1]\big)$:

$$\frac{41}{9} - \frac{F_x^1(\mathbf{0})(\sigma)^2}{G^1(\mathbf{1})(\sigma)^2} = \frac{41}{9} - \frac{F_x^2(\mathbf{0})(\sigma)^2}{G^2(\mathbf{1})(\sigma)^2} = \frac{41}{9}, \qquad \frac{41}{9} - \frac{F_x^3(\mathbf{0})(\sigma)^2}{G^3(\mathbf{1})(\sigma)^2} = \frac{37}{9}, \qquad \ldots$$

Finally, this sequence converges to $41/9 - 25/9 = 16/9$ as the variance of x. △

[7] Here $F_h^k(X)$ stands for k–fold application of F_h to X.

5 Reasoning about Run–Time Variances

In addition to the (co)variance of outcomes we are interested in the variance of the program's *run–time*. We formally capture the run–time variance in terms of an operational model which is given as a Markov Chain (MC for short) with rewards. The model is similar to the ones studied in [6,8], but additionally keeps track of the run–time in a dedicated variable τ which is *not accessible by the program*, but may occur in expressions describing an expectation.

Definition 8 (Run–Time Expectations). *Let* $\mathbb{S}_\tau = \{\sigma \mid \mathbb{V} \uplus \{\tau\} \to \mathbb{Q}\}$*. The* ***set of run–time expectations*** *is then defined as* $\mathbb{E}_{\boldsymbol{\tau}} = \left\{ t \;\middle|\; t\colon \mathbb{S}_\tau \to \mathbb{R}_{\geq 0}^{\infty} \right\}$.

A corresponding wp–style calculus to reason about expected run–times and variances of probabilistic programs is presented afterwards.

We first briefly recall some necessary notions about MCs and refer to [1, Ch. 10] for a comprehensive introduction. A *Markov Chain* is a tuple $\mathcal{M} = (\mathcal{S}, \mathbf{P}, s_I, \mathit{rew})$, where $\mathcal{S}$ is a countable set of *states*, $s_I \in \mathcal{S}$ is the initial state, $\mathbf{P} : \mathcal{S} \times \mathcal{S} \to [0,1]$ is the *transition probability function* such that for each state $s \in \mathcal{S}$, $\sum_{s' \in \mathcal{S}} \mathbf{P}(s, s') \in \{0,1\}$, and $\mathit{rew} : \mathcal{S} \to \mathbb{R}_{\geq 0}$ is a *reward function*. Instead of $\mathbf{P}(s,s') = p$, we often write $s \xrightarrow{p} s'$. A *path* in $\mathcal{M}$ is a finite or infinite sequence $\pi = s_0 s_1 \ldots$ such that $s_i \in S$ and $\mathbf{P}(s_i, s_{i+1}) > 0$ for each $i \geq 0$ (where we tacitly assume $\mathbf{P}(s_i, s_{i+1}) = 0$ if π is a finite path of length n and $i \geq n$). The *cumulative reward* and the probability of a finite path $\hat{\pi} = s_0 \ldots s_n$ are given by $\mathit{rew}(\hat{\pi}) = \sum_{k=0}^{n-1} \mathit{rew}(s_k)$ and $\Pr^{\mathcal{M}}\{\hat{\pi}\} = \prod_{k=0}^{n-1} \mathbf{P}(s_k, s_{k+1})$. These notions are lifted to infinite paths by the standard cylinder set construction (cf. [1]).

Given a set of target states $T \subseteq \mathcal{S}$, $\Diamond T$ denotes the set of all paths in $\mathcal{M}$ reaching a state in T from initial state s_I. Analogously, all paths starting in s_I that never reach a state in T are denoted by $\neg\Diamond T$. The *expected reward* that $\mathcal{M}$ eventually reaches T from a state $s \in \mathcal{S}$ is defined as follows:

$$\mathsf{ExpRew}^{\mathcal{M}}\left(\Diamond T\right) \;=\; \begin{cases} \sum_{\pi \in \Diamond T} \Pr^{\mathcal{M}}\{\pi\} \cdot \mathit{rew}(\pi) & \text{if } \sum_{\pi \in \Diamond T} \Pr^{\mathcal{M}}\{\pi\} = 1 \\ \infty & \text{if } \sum_{\pi \in \Diamond T} \Pr^{\mathcal{M}}\{\pi\} < 1. \end{cases}$$

Moreover, the *conditional expected reward* of $\mathcal{M}$ reaching T from s under the condition that a set of undesired states $U \subseteq \mathcal{S}$ is never reached is given by[8]

$$\mathsf{CExpRew}^{\mathcal{M}}\left(\Diamond T \mid \neg\Diamond U\right) \;=\; \frac{\mathsf{ExpRew}^{\mathcal{M}}\left(\Diamond T \cap \neg\Diamond U\right)}{\Pr^{\mathcal{M}}\{\neg\Diamond U\}}.$$

We are now in a position to define an operational model for our probabilistic programming language $\mathbb{P}$. Let $\downarrow$ and $\lightning$ be two special symbols denoting successful termination of a program and failure of an observation, respectively.

Definition 9 (The Operational MC of a $\mathbb{P}$–Program). *Given a program* $C \in \mathbb{P}$*, an initial program state* $\sigma_0 \in \mathbb{S}_\tau$ *and a post–run–time* $t \in \mathbb{E}$*, the according* MC *is given by* $\mathcal{M}_{\sigma_0}^{t}\left[C\right] = (\mathcal{S},\, \mathbf{P},\, s_I,\, \mathit{rew})$*, where*

[8] Again, we stick to the convention that $\frac{0}{0} = 0$.

$$\frac{}{\langle \downarrow, \sigma \rangle \xrightarrow{1} \langle \mathrm{sink} \rangle}\ [\mathsf{terminated}] \qquad \frac{}{\langle \mathrm{sink} \rangle \xrightarrow{1} \langle \mathrm{sink} \rangle}\ [\mathsf{sink}]$$

$$\frac{}{\langle \texttt{empty}, \sigma \rangle \xrightarrow{1} \langle \downarrow, \sigma \rangle}\ [\mathsf{empty}] \qquad \frac{}{\langle \texttt{skip}, \sigma \rangle \xrightarrow{1} \langle \downarrow, \sigma[\tau/\tau+1] \rangle}\ [\mathsf{skip}]$$

$$\frac{}{\langle \texttt{halt}, \sigma \rangle \xrightarrow{1} \langle \mathrm{sink} \rangle}\ [\mathsf{halt}] \qquad \frac{}{\langle x := E, \sigma \rangle \xrightarrow{1} \langle \downarrow, \sigma[x/E,\ \tau/\tau+1] \rangle}\ [\mathsf{assgn}]$$

$$\frac{\langle C_1, \sigma \rangle \xrightarrow{p} \langle C_1', \sigma' \rangle \quad 0 < p \leq 1}{\langle C_1 ;\ C_2, \sigma \rangle \xrightarrow{p} \langle C_1' ;\ C_2, \sigma' \rangle}\ [\mathsf{seq\text{-}1}] \qquad \frac{}{\langle \downarrow;\ C_2, \sigma \rangle \xrightarrow{1} \langle C_2, \sigma \rangle}\ [\mathsf{seq\text{-}2}]$$

$$\frac{}{\langle \{C_1\}\ [p]\ \{C_2\}, \sigma \rangle \xrightarrow{p} \langle C_1, \sigma[\tau/\tau+1] \rangle}\ [\mathsf{pc\text{-}1}]$$

$$\frac{}{\langle \{C_1\}\ [p]\ \{C_2\}, \sigma \rangle \xrightarrow{1-p} \langle C_2, \sigma[\tau/\tau+1] \rangle}\ [\mathsf{pc\text{-}2}]$$

$$\frac{[B](\sigma) = 1}{\langle \texttt{if}\ (B)\ \{C_1\}\ \texttt{else}\ \{C_2\}, \sigma \rangle \xrightarrow{1} \langle C_1, \sigma[\tau/\tau+1] \rangle}\ [\mathsf{if\text{-}true}]$$

$$\frac{[B](\sigma) = 0}{\langle \texttt{if}\ (B)\ \{C_1\}\ \texttt{else}\ \{C_2\}, \sigma \rangle \xrightarrow{1} \langle C_2, \sigma[\tau/\tau+1] \rangle}\ [\mathsf{if\text{-}false}]$$

$$\frac{}{\langle \texttt{while}\ (B)\ \{C\}, \sigma \rangle \xrightarrow{1} \langle \texttt{if}\ (B)\ \{C;\ \texttt{while}\ (B)\ \{C\}\}\ \texttt{else}\ \{\texttt{empty}\}, \sigma \rangle}\ [\mathsf{while}]$$

$$\frac{}{\langle \texttt{diverge}, \sigma \rangle \xrightarrow{1} \langle \texttt{diverge}, \sigma \rangle}\ [\mathsf{diverge}]$$

$$\frac{[B](\sigma) = 1}{\langle \texttt{observe}\ B, \sigma \rangle \xrightarrow{1} \langle \downarrow, \sigma[\tau/\tau+1] \rangle}\ [\mathsf{observe\text{-}true}]$$

$$\frac{[B](\sigma) = 0}{\langle \texttt{observe}\ B, \sigma \rangle \xrightarrow{1} \langle \lightning \rangle}\ [\mathsf{observe\text{-}false}] \qquad \frac{}{\langle \lightning \rangle \xrightarrow{1} \langle \mathrm{sink} \rangle}\ [\mathsf{observe\text{-}failed}]$$

Fig. 2. Rules for defining the transition probability function of the MC of a $\mathbb{P}$–program.

- $\mathcal{S} = ((\mathbb{P} \cup \{\downarrow\} \cup \{\downarrow; C \mid C \in \mathbb{P}\}) \times \mathbb{S}_\tau) \ \cup\ \{\langle \mathrm{sink} \rangle, \langle \lightning \rangle\}$,
- *the transition probability function* $\mathbf{P}$ *is given by the rules in Figure 2,*
- $s_I = \langle C, \sigma_0 \rangle$, *and*
- $rew : \mathcal{S} \to \mathbb{R}_{\geq 0}$ *is the reward function defined by* $rew(s) = t(\sigma)$ *if* $s = \langle \downarrow, \sigma \rangle$ *for some* $\sigma \in \mathbb{S}_\tau$ *and* $rew(s) = 0$, *otherwise.*

In this construction, $\sigma_0(\tau)$ represents the *post–execution time* of a program, i.e. the run–time that is added after a program finishes its execution. Hence, τ precisely captures the run–time of a program if $\sigma_0(\tau) = 0$. The rules presented in Figure 2 defining the transition probability function are mostly self–explanatory. We assume guard evaluations, probabilistic choices, assignments and the statement `skip` to consume one unit of time. Hence, τ is incremented accordingly for each of these statements and remains untouched otherwise.

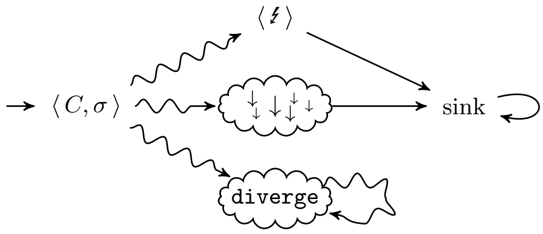


Fig. 3. Schematic depiction of the structure of the operational MC $\mathcal{M}_\sigma^t\,[C]$.

Figure 3 sketches the structure of the operational MC $\mathcal{M}_\sigma^t\,[C]$. Here, clouds represent a set of states and squiggly arrows indicate that a set of states is reachable by one or more paths. Each run either terminates successfully (i.e. it visits some state $\langle \downarrow,\ \sigma' \rangle$), or violates an observation (i.e. it visits $\langle \lightning \rangle$), or diverges. In the first two cases each run eventually ends up in the $\langle \text{sink} \rangle$ state. Note that states of the form $\langle \downarrow,\ \sigma' \rangle$ are the only ones that may have a positive reward. Furthermore, each of the auxiliary states of the form $\langle \downarrow,\ \sigma' \rangle$, $\langle \lightning \rangle$ and $\langle \text{sink} \rangle$ is needed to properly deal with `diverge`, `halt` and `observe` B.

Since τ precisely captures the run–time of a program if τ is initially set to 0, the *expected run–time* of executing $C \in \mathbb{P}$ on input $\sigma \in \mathbb{S}_\tau$ with $\sigma(\tau) = 0$ is given by the conditional expected reward of $\mathcal{M}_\sigma^\tau\,[C]$ reaching $\langle \text{sink} \rangle$, given that no observation fails, i.e. $\mathsf{E}_{[\![C]\!](\sigma)}\,(\tau) = \mathsf{CExpRew}^{\mathcal{M}_\sigma^\tau[C]}\,(\Diamond\langle \text{sink} \rangle \mid \neg\Diamond\langle \lightning \rangle)$. Then, in compliance with Definition 4, the *run–time variance* of $C \in \mathbb{P}$ in state $\sigma \in \mathbb{S}_\tau$ with $\sigma(\tau) = 0$ is given by

$$\begin{aligned}
\mathsf{RTVar}_{[\![C]\!](\sigma)} \;&=\; \mathsf{E}_{[\![C]\!](\sigma)}\left(\tau^2\right) \;-\; \left(\mathsf{E}_{[\![C]\!](\sigma)}\,(\tau)\right)^2 \\
&=\; \mathsf{CExpRew}^{\mathcal{M}_\sigma^{\tau^2}[C]}\,(\Diamond\langle \text{sink} \rangle \mid \neg\Diamond\langle \lightning \rangle) \\
&\qquad -\; \left(\mathsf{CExpRew}^{\mathcal{M}_\sigma^{\tau}[C]}\,(\Diamond\langle \text{sink} \rangle \mid \neg\Diamond\langle \lightning \rangle)\right)^2 .
\end{aligned}$$

In the following we provide a corresponding wp–style calculus to reason about expected run–times and run–time variances of probabilistic programs. A formal definition of the **run–time transformer** $\mathsf{rt}\colon \mathbb{P} \to (\mathbb{E}_\tau \to \mathbb{E}_\tau)$ is provided in Table 1 (rightmost column). Intuitively, it behaves like wp except that a *dedicated run–time variable* τ is updated accordingly for each program statement that consumes time. In [8], a transformer for expected run–times without the need for an additional variable τ is studied. However, this approach fails when reasoning about run–time variances since it fails to capture expected squared run–times. The run–time transformer rt precisely captures the notion of expected run–time of our operational model.

Theorem 6 (Operational–Denotational Correspondence). *Let $C \in \mathbb{P}$, $t \in \mathbb{E}_\tau$, and $\sigma \in \mathbb{S}_\tau$. Then*

$$\mathsf{CExpRew}^{\mathcal{M}_\sigma^{t}[C]}\,(\Diamond\langle \text{sink} \rangle \mid \neg\Diamond\langle \lightning \rangle) \;=\; \frac{\mathsf{rt}\,[C]\,(t)\,(\sigma)}{\mathsf{wlp}\,[C]\,(1)\,(\sigma)}.$$

Proof. By structural induction on $C \in \mathbb{P}$. See Appendix A.7. □

As a result of Theorem 6 we immediately obtain a formal definition of the run–time variance of probabilistic programs in terms of rt and wlp. Formally, the *run–time variance* of $C \in \mathbb{P}$ in state $\sigma \in \mathbb{S}_\tau$ with $\sigma(\tau) = 0$ is given by

$$\begin{aligned}\mathsf{RTVar}_{[\![C]\!](\sigma)} &= \mathsf{CExpRew}^{\mathcal{M}_\sigma^{\tau^2}[C]}\left(\Diamond\langle\,\mathrm{sink}\,\rangle \mid \neg\Diamond\langle\,\lightning\,\rangle\right) \\ &\quad - \left(\mathsf{CExpRew}^{\mathcal{M}_\sigma^{\tau}[C]}\left(\Diamond\langle\,\mathrm{sink}\,\rangle \mid \neg\Diamond\langle\,\lightning\,\rangle\right)\right)^2 \\ &= \frac{\mathsf{rt}\,[C]\,(\tau^2)\,(\sigma)}{\mathsf{wlp}\,[C]\,(\mathbf{1})\,(\sigma)} - \frac{(\mathsf{rt}\,[C]\,(\tau)\,(\sigma))^2}{(\mathsf{wlp}\,[C]\,(\mathbf{1})\,(\sigma))^2}.\end{aligned}$$

Since rt is continuous (cf. Appendix A.8 for a formal proof), the invariant–aided approach based on Park's Lemma (Theorem 4) presented in Section 4 is applicable to approximate run–time variances as well. We present the formal result for approximating upper bounds only. The dual result for lower bounds is obtained analogously.

Theorem 7 (Invariant–Aided Over–Approximation of Run–Time Variances). *Let $C = \texttt{while}\ (B)\ \{C'\}$ and $\sigma \in \mathbb{S}_\tau$ with $\sigma(\tau) = 0$. Moreover, let $F_h(X) = [\neg B] \cdot h + [B] \cdot \mathsf{rt}\,[C']\,(X)$, and $G(Y) = [\neg B] + [B] \cdot \mathsf{wlp}\,[C']\,(Y)$. Furthermore, let $\widehat{X} \in \mathbb{E}_\tau$ and $\widehat{Y} \in \mathbb{E}_{\leq 1}$, such that $F_{\tau^2}(\widehat{X}) \preceq \widehat{X}$, $\widehat{Y} \preceq G(\widehat{Y})$, and $\widehat{Y}(\sigma) > 0$. Then for each $k \in \mathbb{N}$, it holds*

$$\mathsf{RTVar}_{[\![C]\!](\sigma)} \leq \frac{\widehat{X}(\sigma)}{\widehat{Y}(\sigma)} - \left(\frac{F_\tau^k(\mathbf{0})(\sigma)}{G^k(\mathbf{1})(\sigma)}\right)^2.$$

The proof of Theorem 7 is analogous to the proof of Theorem 5. Again, since it is always possible to choose $\widehat{X} = \mathsf{lfp}\,F_{\tau^2}$ and $\widehat{Y} = \mathsf{gfp}\,G$, Theorem 7 is complete, i.e. there exist $\widehat{X} \in \mathbb{E}_\tau$ and $\widehat{Y} \in \mathbb{E}_{\leq 1}$ such that

$$\inf_{k \in \mathbb{N}} \frac{\widehat{X}(\sigma)}{\widehat{Y}(\sigma)} - \left(\frac{F_\tau^k(\mathbf{0})(\sigma)}{G^k(\mathbf{1})(\sigma)}\right)^2 = \mathsf{RTVar}_{[\![C]\!](\sigma)}.$$

6 Conclusion

We have studied the computational hardness of obtaining both upper and lower bounds on (co)variance of outcomes and established that this is Σ_2^0–complete. Thus neither upper nor lower bounds are computably enumerable. Furthermore, we have established that deciding whether the (co)variance equals a given rational and deciding whether the covariance is infinite is Π_2^0–complete.

In the second part of the paper, we continued by presenting a sound and complete invariant–aided approach which allows to computably enumerate upper and lower bounds on (co)variances of while–loops, once appropriate loop–invariants are found. Finally, we have shown how this approach can be extended to reason about the variance of run–times.

References

1. Baier, C., Katoen, J.P.: Principles of Model Checking. MIT Press (2008)
2. Davis, M.D.: Computability, Complexity, and Languages: Fundamentals of Theoretical Computer Science. Academic Press (1994)
3. Dijkstra, E.W.: A Discipline of Programming. Prentice-Hall Englewood Cliffs (1976)
4. Fioriti, L.M.F., Hermanns, H.: Probabilistic termination: Soundness, completeness, and compositionality. In: POPL 2015. pp. 489–501. ACM (2015)
5. Gordon, A.D., Henzinger, T.A., Nori, A.V., Rajamani, S.K.: Probabilistic programming. In: Future of Software Engineering (FOSE). pp. 167–181. ACM (2014)
6. Jansen, N., Kaminski, B.L., Katoen, J.P., Olmedo, F., Gretz, F., McIver, A.: Conditioning in probabilistic programming. ENTCS 319, 199–216 (2015)
7. Kaminski, B.L., Katoen, J.P.: On the hardness of almost-sure termination. In: Proc. of MFCS 2015, Part I. LNCS, vol. 9234, pp. 307–318. Springer (2015)
8. Kaminski, B.L., Katoen, J.P., Matheja, C., Olmedo, F.: Weakest precondition reasoning for expected run–times of probabilistic programs. In: Proc. of ESOP 2016 (2016), (to appear)
9. Kleene, S.C.: Recursive Predicates and Quantifiers. Trans. of the AMS 53(1), 41 – 73 (1943)
10. McIver, A., Morgan, C.: Abstraction, Refinement and Proof for Probabilistic Systems. Springer (2004)
11. Odifreddi, P.: Classical Recursion Theory: The Theory of Functions and Sets of Natural Numbers. Elsevier (1992)
12. Odifreddi, P.: Classical Recursion Theory, Volume II. Elsevier (1999)
13. Post, E.L.: Recursively Enumerable Sets of Positive Integers and their Decision Problems. Bulletin of the AMS 50(5), 284–316 (1944)
14. Wechler, W.: Universal Algebra for Computer Scientists, EATCS Monographs on Theoretical Computer Science, vol. 25. Springer (1992)

A Appendix

A.1 Remaining Proof of Lemma 1

For proving $\mathcal{RCOVAR} \in \Sigma_2^0$, consider that $(C,\, \sigma,\, f,\, g,\, q) \in \mathcal{RCOVAR}$ iff

$$\exists\, \delta > 0\; \exists\, \ell\; \forall\, k\colon\; \frac{\mathsf{wp}^k\,[C]\,(f \cdot g)\,(\sigma)}{\mathsf{wlp}^k\,[C]\,(\mathbf{1})\,(\sigma)} - \frac{\mathsf{wp}^\ell\,[C]\,(f)\,(\sigma) \cdot \mathsf{wp}^\ell\,[C]\,(g)\,(\sigma)}{\mathsf{wlp}^\ell\,[C]\,(\mathbf{1})\,(\sigma)^2} \;<\; q - \delta\;.$$

□

A.2 Proof of Lemma 2

For proving $\mathcal{COVAR} \in \Sigma_2^0$, consider that $(C,\, \sigma,\, f,\, g,\, q) \in \mathcal{COVAR}$ iff

$$(C,\, \sigma,\, f,\, g,\, q) \notin \mathcal{LCOVAR} \quad \text{and} \quad (C,\, \sigma,\, f,\, g,\, q) \notin \mathcal{RCOVAR}\;.$$

Now by Lemma 1 there must exist *decidable* relations L and R, such that $(C,\, \sigma,\, f,\, g,\, q) \in \mathcal{COVAR}$ iff

$$\begin{aligned}
& \neg\exists\, y_1\, \forall\, y_2\colon (y_1,\, y_2,\, C, \sigma, f, g, q) \in L \;\wedge\; \neg\exists\, y_1'\, \forall\, y_2'\colon (y_1',\, y_2',\, C, \sigma, f, g, q) \in R \\
\iff\; & \forall\, y_1\, \exists\, y_2\colon (y_1,\, y_2,\, C, \sigma, f, g, q) \notin L \;\wedge\; \forall\, y_1'\, \exists\, y_2'\colon (y_1',\, y_2',\, C, \sigma, f, g, q) \notin R \\
\iff\; & \forall\, y_1\, \forall\, y_1'\, \exists\, y_2\, \exists\, y_2'\colon (y_1,\, y_2,\, C, \sigma, f, g, q) \notin L \;\wedge\; (y_1',\, y_2',\, C, \sigma, f, g, q) \notin R\;,
\end{aligned}$$

which is a Π_2–formula.

For proving ${}^\infty\mathcal{COVAR} \in \Pi_2^0$, consider that $(C,\, \sigma,\, f,\, g,\, q) \in {}^\infty\mathcal{COVAR}$ iff

$$\neg\; \exists\, b_1\, \exists\, b_2\, \forall\, k\colon\, b_1 \leq \frac{\mathsf{wp}^k\,[C]\,(f \cdot g)\,(\sigma)}{\mathsf{wlp}^k\,[C]\,(\mathbf{1})\,(\sigma)} - \frac{\mathsf{wp}^k\,[C]\,(f)\,(\sigma) \cdot \mathsf{wp}^k\,[C]\,(g)\,(\sigma)}{\mathsf{wlp}^k\,[C]\,(\mathbf{1})\,(\sigma)^2} \leq b_2\;.$$

The above is the negation of a Σ_2^0–formula which is equivalent to a Π_2^0–formula.

□

A.3 Remaining Proof of Lemma 3

For proving the Σ_2^0–hardness of $\mathcal{RCOVAR}$, we reduce the Σ_2^0–complete $\overline{\mathcal{AST}}$ to $\mathcal{RCOVAR}$. For that, consider the reduction function $r_{\mathcal{R}}(C,\, \sigma) = \big(C',\, \sigma,\, v,\, v,\, \frac{1}{4}\big)$, with C' given by

$$v \coloneqq 0\,;\; \{\texttt{diverge}\}\; [1/2]\; \{C\}\,;\; v \coloneqq 1\;,$$

where variable v does not occur in C. We have

$$\mathsf{Cov}_{[\![C']\!](\sigma)}\,(v,\, v) \;\;=\;\; \mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma) - \mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma)^2\;.$$

Recall that $\mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma)$ is exactly the probability of C' terminating on input σ. By reconsidering Figure 1, we can see that $\mathsf{Cov}_{[\![C']\!](\sigma)}\,(v,\, v) = \mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma) - \mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma)^2 < \frac{1}{4}$ holds iff C' does not terminate with probability $1/2$. Since by construction C' terminates with a probability of at most $1/2$, it follows that $\mathsf{Cov}_{[\![C']\!](\sigma)}\,(v,\, v) < \frac{1}{4}$ holds iff C' terminates with probability less than 1, which is the case iff C terminates with probability less than 1. Thus $r_{\mathcal{R}}(C,\, \sigma) = \big(C',\, \sigma,\, v,\, v,\, \frac{1}{4}\big) \in \mathcal{RCOVAR}$ iff $(C,\, \sigma) \in \overline{\mathcal{AST}}$ and therefore we have $r_{\mathcal{R}}\colon \overline{\mathcal{AST}} \leq_{\mathrm{m}} \mathcal{RCOVAR}$. Since $\overline{\mathcal{AST}}$ is Σ_2^0–complete, if follows that $\mathcal{RCOVAR}$ is Σ_2^0–hard.

□

A.4 Proof of Lemma 4

For proving the Π^0_2–hardness of $\mathcal{COVAR}$, we reduce the Π^0_2–complete $\mathcal{AST}$ to $\mathcal{COVAR}$. For that, consider the reduction function $r_{\mathcal{V}}(C,\,\sigma) = \big(C',\,\sigma,\,v,\,v,\,\frac{1}{4}\big)$, with C' given by

$$v \coloneqq 0\texttt{;}\ \{\texttt{diverge}\}\ [1/2]\ \{C\}\texttt{;}\ v \coloneqq 1\ ,$$

where variable v does not occur in C. Again, we have

$$\mathsf{Cov}_{\llbracket C' \rrbracket(\sigma)}\,(v,\,v) \quad = \quad \mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma) - \mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma)^2$$

(cf. proof of Lemma 3). A plot of the latter is given in Figure 1. Recall that $\mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma)$ is exactly the probability of C' terminating on input σ. We can see that $\mathsf{Cov}_{\llbracket C' \rrbracket(\sigma)}\,(v,\,v) = \mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma) - \mathsf{wp}\,[C']\,(\mathbf{1})\,(\sigma)^2 = \frac{1}{4}$ iff C' terminates with probability $1/2$. Since C' terminates at most with probability $1/2$, we obtain that $\mathsf{Cov}_{\llbracket C' \rrbracket(\sigma)}\,(v,\,v) = \frac{1}{4}$ iff C' terminates with probability $1/2$, which is the case iff C terminates almost–surely. Thus $r_{\mathcal{V}}(C,\,\sigma) = \big(C',\,\sigma,\,v,\,v,\,\frac{1}{4}\big) \in \mathcal{COVAR}$ iff $(C,\,\sigma) \in \mathcal{AST}$ and therefore $r_{\mathcal{V}}\colon \mathcal{AST} \leq_{\mathrm{m}} \mathcal{COVAR}$. Since $\mathcal{AST}$ is Π^0_2–complete, we obtain that $\mathcal{COVAR}$ is Π^0_2–hard. □

A.5 Proof of Theorem 5

By the precondition of Theorem 5 and by Theorem 4, $\mathsf{wp}\,[C]\,(f \cdot g)\,(\sigma) \leq \widehat{X}(\sigma)$ and $0 < \widehat{Y}(\sigma) \leq \mathsf{wlp}\,[C]\,(\mathbf{1})\,(\sigma)$. Therefore

$$\mathsf{E}_{\llbracket C \rrbracket(\sigma)}\,(f \cdot g) \quad = \quad \frac{\mathsf{wp}\,[C]\,(f \cdot g)\,(\sigma)}{\mathsf{wlp}\,[C]\,(\mathbf{1})\,(\sigma)} \quad \leq \quad \frac{\widehat{X}(\sigma)}{\widehat{Y}(\sigma)}\ .$$

Furthermore, for each $k \in \mathbb{N}$ we have

$$\begin{aligned} \mathsf{wp}^k\,[C]\,(f)\,(\sigma) &\leq \mathsf{wp}\,[C]\,(f)\,(\sigma)\ , \\ \mathsf{wp}^k\,[C]\,(g)\,(\sigma) &\leq \mathsf{wp}\,[C]\,(g)\,(\sigma)\ , \qquad \text{and} \\ \mathsf{wp}^k\,[C]\,(\mathbf{1})\,(\sigma) &\geq \mathsf{wlp}\,[C]\,(\mathbf{1})\,(\sigma)\ , \end{aligned}$$

which all together yields

$$\begin{aligned} &\frac{\mathsf{wp}^k\,[C]\,(f)\,(\sigma) \cdot \mathsf{wp}^k\,[C]\,(g)\,(\sigma)}{\mathsf{wlp}^k\,[C]\,(\mathbf{1})\,(\sigma)^2} \\ &\qquad \leq \quad \frac{\mathsf{wp}\,[C]\,(f)\,(\sigma) \cdot \mathsf{wp}\,[C]\,(g)\,(\sigma)}{\mathsf{wlp}\,[C]\,(\mathbf{1})\,(\sigma)^2} \quad = \quad \mathsf{E}_{\llbracket C \rrbracket(\sigma)}\,(f) \cdot \mathsf{E}_{\llbracket C \rrbracket(\sigma)}\,(g) \end{aligned}$$

An over–approximation of $\mathsf{E}_{\llbracket C \rrbracket(\sigma)}\,(f \cdot g)$ subtracted by an under–approximation of $\mathsf{E}_{\llbracket C \rrbracket(\sigma)}\,(f) \cdot \mathsf{E}_{\llbracket C \rrbracket(\sigma)}\,(g)$ yields then an over–approximation of $\mathsf{E}_{\llbracket C \rrbracket(\sigma)}\,(f \cdot g) - \mathsf{E}_{\llbracket C \rrbracket(\sigma)}\,(f) \cdot \mathsf{E}_{\llbracket C \rrbracket(\sigma)}\,(g) = \mathsf{Cov}_{\llbracket C \rrbracket(\sigma)}\,(f,\,g)$. □

A.6 Invariant–Aided Under–Approximation of Covariances

Theorem 8 (Invariant–Aided Under–Approximation of Covariances). *Let* $C = \texttt{while}\ (B)\ \{C'\}$*,* $\sigma \in \mathbb{S}$*,* $f, g \in \mathbb{E}$*,* $F_h(X) = [\neg B] \cdot h + [B] \cdot \mathsf{wp}\,[C']\,(X)$*, and* $G(Y) = [\neg B] + [B] \cdot \mathsf{wlp}\,[C']\,(Y)$*. Furthermore, let* $\widehat{X}_f, \widehat{X}_g \in \mathbb{E}$ *and* $\widehat{Y} \in \mathbb{E}_{\leq 1}$*, such that* $F_f(\widehat{X}_f) \preceq \widehat{X}_f$*,* $F_g(\widehat{X}_g) \preceq \widehat{X}_g$*, and* $\widehat{Y} \preceq G(\widehat{Y})$*. Then for all* $k \in \mathbb{N}$ *it holds that*

$$\frac{\mathsf{wp}^k\,[C]\,(f \cdot g)\,(\sigma)}{\mathsf{wlp}^k\,[C]\,(\mathbf{1})\,(\sigma)} - \frac{\widehat{X}_f(\sigma) \cdot \widehat{X}_g(\sigma)}{\widehat{Y}(\sigma)^2} \;\leq\; \mathsf{Cov}_{[\![C]\!](\sigma)}\,(f,\, g)\ .$$

Proof. Analogous to the proof of Theorem 5.

Corollary 2 (Completeness of Theorem 8). *Let* $C = \texttt{while}\ (B)\ \{C'\}$*,* $\sigma \in \mathbb{S}$*,* $f, g \in \mathbb{E}$*. Then there exist* $\widehat{X}_f, \widehat{X}_g \in \mathbb{E}$ *and* $\widehat{Y} \in \mathbb{E}_{\leq 1}$*, such that*

$$\sup_{k \in \mathbb{N}} \frac{\mathsf{wp}^k\,[C]\,(f \cdot g)\,(\sigma)}{\mathsf{wlp}^k\,[C]\,(\mathbf{1})\,(\sigma)} - \frac{\widehat{X}_f(\sigma) \cdot \widehat{X}_g(\sigma)}{\widehat{Y}(\sigma)^2} \;=\; \mathsf{Cov}_{[\![C]\!](\sigma)}\,(f,\, g)\ .$$

A.7 Proof of Theorem 6

The proof relies on several auxiliary results which are presented first.

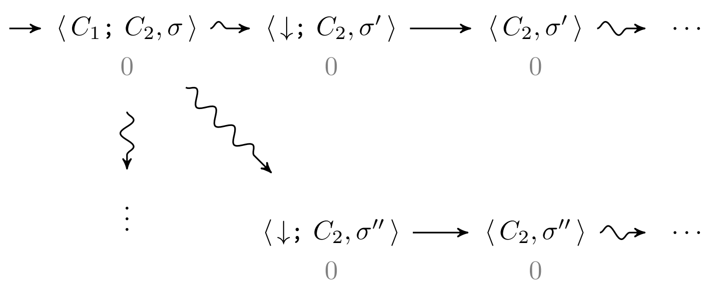
Lemma 6. *Let* $C_1, C_2 \in \mathbb{P}$*,* $f \in \mathbb{E}$*, and* $\sigma \in \mathbb{S}_\tau$*. Then*

$$\mathit{ExpRew}^{\mathcal{M}_\sigma^f[C_1;\ C_2]}\,(\Diamond\langle\,\mathrm{sink}\,\rangle) \;=\; \mathit{ExpRew}^{\mathcal{M}_\sigma^{g(C_2,f)}[C_1]}\,(\Diamond\langle\,\mathrm{sink}\,\rangle)\,,$$

where

$$g(C_2, f) \;=\; \mathit{ExpRew}^{\lambda\rho\,.\,\mathcal{M}_\rho^f[C_2]}\,(\Diamond\langle\,\mathrm{sink}\,\rangle)\,.$$

Proof. The MC $\mathcal{M}_\sigma^f\,[C]$ is of the following form:



Hence, every path starting in $\langle C_1\,;\; C_2, \sigma \rangle$ either eventually reaches a state $\langle \downarrow;\; C_2, \sigma' \rangle$ for some $\sigma' \in \mathbb{S}_\tau$ and then immediately reaches $\langle C_2, \sigma' \rangle$ or diverges, i.e. never reaches $\langle \text{sink} \rangle$. Since $\langle C_2, \sigma' \rangle$ is the initial state of MC $\mathcal{M}_{\sigma'}^{f}[C_2]$, we can transform $\mathcal{M}_{\sigma}^{f}[C_1\,;\; C_2]$ into an MC $\mathcal{M}_{\sigma}^{g(C_2,f)}[C_1]$ having the same expected reward by setting

$$g(C_2, f) \;=\; \mathsf{ExpRew}^{\lambda\rho\,.\,\mathcal{M}_{\rho}^{f}[C_2]}(\Diamond\langle \text{sink} \rangle)\,.$$

Definition 10 (Bounded `while` Loops). *Let $C \in \mathbb{P}$, $f \in \mathbb{E}$, and $k \in \mathbb{N}$. Then*

$$\begin{aligned} \texttt{while}^{<0}\;(B)\;\{C\} \;&=\; \texttt{halt},\; \textit{and} \\ \texttt{while}^{<k+1}\;(B)\;\{C\} \;&=\; \texttt{if}\;(B)\;\big\{C\texttt{;}\;\texttt{while}^{<k}\;(B)\;\{C\}\big\}\;\texttt{else}\;\{\texttt{empty}\} \end{aligned}$$

To improve readability, let $C' = \texttt{while}\;(B)\;\{C\}$ and $C_k = \texttt{while}^{<k}\;(B)\;\{C\}$ for the remainder of this section.

Lemma 7. *Let $C \in \mathbb{P}$ and $f \in \mathbb{E}$. Then*

$$\sup_{k \in \mathbb{N}} \mathsf{rt}\left[\texttt{while}^{<k}\;(B)\;\{C\}\right](f) \;=\; \mathsf{rt}\left[\texttt{while}\;(B)\;\{C\}\right](f)\,.$$

Proof. Let $F_f(X) \;=\; \big([\neg B]\cdot f + [B]\cdot \mathsf{rt}[C](X)\big)[\tau/\tau+1]$. Assume, for the moment, that for each $k \in \mathbb{N}$, we have $\mathsf{rt}[C_k](f) = F_f^k(\mathsf{0})$. Then, using Kleene's Fixed Point Theorem, we can establish that

$$\sup_{k\in\mathbb{N}} \mathsf{rt}[C_k](f) \;=\; \sup_{k\in\mathbb{N}} F_f^k(\mathsf{0}) \;=\; \mathsf{lfp}\, X.\; F_f(X) \;=\; \mathsf{rt}[C'](f)\,.$$

Hence, it suffices to show $\mathsf{rt}[C_k](f) = F_f^k(\mathsf{0})$ for each $k \in \mathbb{N}$. We proceed by induction on k.

I.B. $k = 0$

$$\mathsf{rt}[C_0]((\,)\,f) \;=\; \mathsf{rt}[\texttt{halt}](f) \;=\; \mathsf{0} \;=\; F_f^0(\mathsf{0}).$$

I.H. Assume that $\mathsf{rt}[C_k](f) = F_f^k(\mathsf{0})$ holds for an arbitrary, fixed $k \in \mathbb{N}$.

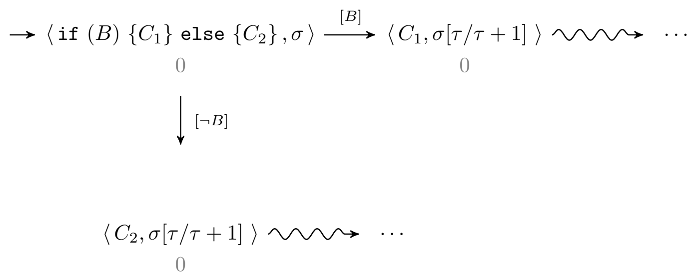
I.S. $k \mapsto k+1$

$$\begin{aligned} & \mathsf{rt}[C_{k+1}](f) \\ =\;& \mathsf{rt}[\texttt{if}\;(B)\;\{C\texttt{;}\;C_k\}\;\texttt{else}\;\{\texttt{empty}\}](f) && \text{(Def. } C_{k+1}\text{)} \\ =\;& \big([\neg B]\cdot f + [B]\cdot \mathsf{rt}[C](\mathsf{rt}[C_k](f))\big)[\tau/\tau+1] && \text{(Table 1)} \\ =\;& \big([\neg B]\cdot f + [B]\cdot \mathsf{rt}[C]\big(F_f^k(\mathsf{0})\big)\big)[\tau/\tau+1] && \text{(I.H.)} \\ =\;& F_f^{k+1}(\mathsf{0}) && \text{(Def. } F_f\text{)} \end{aligned}$$

Lemma 8. *Let $C_1, C_2 \in \mathbb{P}$, $f \in \mathbb{E}$ and $\sigma \in \mathbb{S}_\tau$. Then*

$$\begin{aligned} & \mathit{ExpRew}^{\mathcal{M}_\sigma^f[\texttt{if}\,(B)\,\{C_1\}\,\texttt{else}\,\{C_2\}]}\,(\lozenge\langle\,\mathrm{sink}\,\rangle) \\ = \ & [B](\sigma)\cdot \mathit{ExpRew}^{\mathcal{M}^f_{\sigma[\tau/\tau+1]}[C_1]}\,(\lozenge\langle\,\mathrm{sink}\,\rangle) \\ & + [\neg B](\sigma)\cdot \mathit{ExpRew}^{\mathcal{M}^f_{\sigma[\tau/\tau+1]}[C_2]}\,(\lozenge\langle\,\mathrm{sink}\,\rangle)\,. \end{aligned}$$

Proof. As shown in the figure below, two cases arise.



If $[B](\sigma) = 1$ then each path $\pi \in \lozenge\langle\,\mathrm{sink}\,\rangle$ reaches $\langle\, C_1, \sigma[\tau/\tau+1]\,\rangle$ with probability one. Conversely, if $[B](\sigma) = 1$ then each path $\pi \in \lozenge\langle\,\mathrm{sink}\,\rangle$ reaches $\langle\, C_2, \sigma[\tau/\tau+1]\,\rangle$ with probability one. These states are the initial states of the MCs $\mathcal{M}^f_{\sigma[\tau/\tau+1]}[C_1]$ and $\mathcal{M}^f_{\sigma[\tau/\tau+1]}[C_2]$, respectively.

Lemma 9. *Let $C \in \mathbb{P}$, $f \in \mathbb{E}$ and $\sigma \in \mathbb{S}_\tau$. Then*

$$\sup_{k\in\mathbb{N}} \mathit{ExpRew}^{\mathcal{M}_\sigma^f[\texttt{while}^{<k}\,(B)\,\{C\}]}\,(\lozenge\langle\,\mathrm{sink}\,\rangle) \ \le\ \mathit{ExpRew}^{\mathcal{M}_\sigma^f[\texttt{while}\,(B)\,\{C\}]}\,(\lozenge\langle\,\mathrm{sink}\,\rangle)\,.$$

Proof. We show

$$\mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C_k]}\,(\lozenge\langle\,\mathrm{sink}\,\rangle) \ \le\ \mathsf{ExpRew}^{\mathcal{M}_\sigma^f[\texttt{while}\,(B)\,\{C\}]}\,(\lozenge\langle\,\mathrm{sink}\,\rangle)$$

for each $k \in \mathbb{N}$ by induction on k.

I.B. $k = 0$

$$\begin{aligned} & \mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C_0]}\,(\lozenge\langle\,\mathrm{sink}\,\rangle) \\ = \ & \mathsf{ExpRew}^{\mathcal{M}_\sigma^f[\texttt{halt}]}\,(\lozenge\langle\,\mathrm{sink}\,\rangle) \\ = \ & 0 \le \mathsf{ExpRew}^{\mathcal{M}_\sigma^f[\texttt{while}\,(B)\,\{C\}]}\,(\lozenge\langle\,\mathrm{sink}\,\rangle)\,. \end{aligned}$$

I.H. Assume for an arbitrary, fixed $k \in \mathbb{N}$ that

$$\mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C_k]}\,(\lozenge\langle\,\mathrm{sink}\,\rangle) \ \le\ \mathsf{ExpRew}^{\mathcal{M}_\sigma^f[\texttt{while}\,(B)\,\{C\}]}\,(\lozenge\langle\,\mathrm{sink}\,\rangle)$$

I.S. $k \mapsto k+1$

$$
\begin{aligned}
& \mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C_{k+1}]}(\Diamond\langle\,\text{sink}\,\rangle) \\
=\ & \mathsf{ExpRew}^{\mathcal{M}_\sigma^f[\texttt{if}\,(B)\,\{C;\ C_k\}\,\texttt{else}\,\{\texttt{empty}\}]}(\Diamond\langle\,\text{sink}\,\rangle) && (\text{Def. } C_{k+1}) \\
=\ & [B](\sigma)\cdot \mathsf{ExpRew}^{\mathcal{M}_{\sigma[\tau/\tau+1]}^f[C;\ C_k]}(\Diamond\langle\,\text{sink}\,\rangle) && (\text{Lemma 8}) \\
& + [\neg B](\sigma)\cdot \mathsf{ExpRew}^{\mathcal{M}_{\sigma[\tau/\tau+1]}^f[\texttt{empty}]}(\Diamond\langle\,\text{sink}\,\rangle) \\
=\ & [B](\sigma)\cdot \mathsf{ExpRew}^{\mathcal{M}_{\sigma[\tau/\tau+1]}^{g(C_k,f)}[C]}(\Diamond\langle\,\text{sink}\,\rangle) && (\text{Lemma 6}) \\
& + [\neg B](\sigma)\cdot \mathsf{ExpRew}^{\mathcal{M}_{\sigma[\tau/\tau+1]}^f[\texttt{empty}]}(\Diamond\langle\,\text{sink}\,\rangle) \\
\leq\ & [B](\sigma)\cdot \mathsf{ExpRew}^{\mathcal{M}_{\sigma[\tau/\tau+1]}^{g(C',f)}[C]}(\Diamond\langle\,\text{sink}\,\rangle) && (\text{I.H.}) \\
& + [\neg B](\sigma)\cdot \mathsf{ExpRew}^{\mathcal{M}_{\sigma[\tau/\tau+1]}^f[\texttt{empty}]}(\Diamond\langle\,\text{sink}\,\rangle) \\
=\ & [B](\sigma)\cdot \mathsf{ExpRew}^{\mathcal{M}_{\sigma[\tau/\tau+1]}^f[C;\ C']}(\Diamond\langle\,\text{sink}\,\rangle) && (\text{Lemma 6}) \\
& + [\neg B](\sigma)\cdot \mathsf{ExpRew}^{\mathcal{M}_{\sigma[\tau/\tau+1]}^f[\texttt{empty}]}(\Diamond\langle\,\text{sink}\,\rangle) \\
=\ & \mathsf{ExpRew}^{\mathcal{M}_\sigma^f[\texttt{if}\,(B)\,\{C;\ C'\}\,\texttt{else}\,\{\texttt{empty}\}]}(\Diamond\langle\,\text{sink}\,\rangle) && (\text{Lemma 8}) \\
=\ & \mathsf{ExpRew}^{\mathcal{M}_\sigma^f[\texttt{while}\,(B)\,\{C\}]}(\Diamond\langle\,\text{sink}\,\rangle)\,,
\end{aligned}
$$

where the last step is immediate, because each path of $\mathcal{M}_\sigma^f[C']$ starting in $\langle\, C', \sigma\,\rangle$ (which has 0 reward) first reaches state $\langle\,\texttt{if}\ (B)\ \{C;\ C'\}\ \texttt{else}\ \{\texttt{empty}\}\,, \sigma\,\rangle$ with probability 1.

Lemma 10. *Let $C \in \mathbb{P}$, $f \in \mathbb{E}$ and $\sigma \in \mathbb{S}_\tau$. Then*

$$
\sup_{k\in\mathbb{N}} \mathit{ExpRew}^{\mathcal{M}_\sigma^f[\texttt{while}^{<k}\,(B)\,\{C\}]}(\Diamond\langle\,\text{sink}\,\rangle) \ \geq\ \mathit{ExpRew}^{\mathcal{M}_\sigma^f[\texttt{while}\,(B)\,\{C\}]}(\Diamond\langle\,\text{sink}\,\rangle)\,.
$$

Proof. Let $\pi \in \Diamond\langle\,\text{sink}\,\rangle$ be a path in $\mathcal{M}_\sigma^f[C']$. Then there exists a finite prefix $\hat{\pi}$ of π reaching a state $\langle\,\downarrow, \sigma'\,\rangle$ for some $\sigma' \in \mathbb{S}_\tau$ such that $\mathit{rew}(\pi) = \mathit{rew}(\hat{\pi}) > 0$. Since $\hat{\pi}$ is finite, only finitely many states with first component C', say k, are visited. We show that a corresponding path $\hat{\pi}'$ with $\mathit{rew}(\hat{\pi}') = \mathit{rew}(\hat{\pi})$ exists in $\mathcal{M}_\sigma^f[C_k]$ by induction on $k \geq 1$.

I.B. $k=1$ There exists only one suitable path $\hat{\pi}$ in $\mathcal{M}_\sigma^f[C']$ reaching a state with first component $\downarrow$, which is

$$
\begin{aligned}
\hat{\pi} \ =\ & \langle\, C', \sigma\,\rangle\langle\,\texttt{if}\ (B)\ \{C;\ C'\}\ \texttt{else}\ \{\texttt{empty}\}\,\rangle \\
& \langle\,\texttt{empty}, \sigma[\tau/\tau+1]\,\rangle\langle\,\downarrow, \sigma[\tau/\tau+1]\,\rangle.
\end{aligned}
$$

The corresponding path in $\mathcal{M}_\sigma^f[C_k]$ with the same reward $(f(\sigma[\tau/\tau+1]))$ is

$$
\hat{\pi} \ =\ \langle\, C_1, \sigma\,\rangle\langle\,\texttt{empty}, \sigma[\tau/\tau+1]\,\rangle\langle\,\downarrow, \sigma[\tau/\tau+1]\,\rangle.
$$

I.H. For each finite path $\hat{\pi}$ reaching a state with first component $\downarrow$ and positive reward in $\mathcal{M}_{\sigma}^{f}\,[C']$ visiting $k > 1$ (for an arbitrary, fixed $k \geq 1$) states with first component C', there exists a path $\hat{\pi}'$ reaching a state with first component $\downarrow$ in $\mathcal{M}_{\sigma}^{f}\,[C_k]$ such that $rew(\hat{\pi}) = rew(\hat{\pi}')$.

I.S. $k \mapsto k+1$ Each finite path $\hat{\pi}$ as described above is of the form

$$\begin{aligned}\hat{\pi} \;&=\; \langle\, C', \sigma \,\rangle \langle\, \texttt{if}\ (B)\ \{C\texttt{;}\ C'\}\ \texttt{else}\ \{\texttt{empty}\}\,\rangle \\ &\quad\ \langle\, C\texttt{;}\ C', \sigma' \,\rangle \ldots \langle\, C', \sigma'' \,\rangle \ldots \langle\, \downarrow, \sigma''' \,\rangle\end{aligned}$$

such that k states with first component C' are visited when starting in state $\langle\, C', \sigma' \,\rangle$. Let $\hat{\pi}_2$ be a path starting in this state. By I.H. there exists a corresponding path $\hat{\pi}_2'$ in $\mathcal{M}_{\sigma'}^{f}\,[C_k]$ such that $rew(\hat{\pi}_2) = rew(\hat{\pi}_2')$. Then

$$\hat{\pi}' \;=\; \langle\, C_{k+1}, \sigma \,\rangle \ldots, \langle\, \downarrow\texttt{;}\ C_k, \sigma' \,\rangle \hat{\pi}_2'$$

is a path in $\mathcal{M}_{\sigma'}^{f}\,[C_{k+1}]$ with $rew(\hat{\pi}') = rew(\hat{\pi})$. Hence, for each finite path $\hat{\pi}$ with positive reward in $\mathcal{M}_{\sigma}^{f}\,[C']$ there exists a corresponding path $\hat{\pi}'$ in $\mathcal{M}_{\sigma}^{f}\,[C_k]$ for some $k \in \mathbb{N}$. Thus, we include all paths with positive reward in the MC $\mathcal{M}_{\sigma}^{f}\,[C']$ by taking the supremum over $k \in \mathbb{N}$ of the expected reward of the MCs $\mathcal{M}_{\sigma}^{f}\,[C_k]$.

Putting Lemma 9 and Lemma 10 together, we immediately obtain

Lemma 11. *Let $C \in \mathbb{P}$, $f \in \mathbb{E}$ and $\sigma \in \mathbb{S}_{\tau}$. Then*

$$\sup_{k\in\mathbb{N}} \mathit{ExpRew}^{\mathcal{M}_{\sigma}^{f}\,[\texttt{while}^{<k}\,(B)\,\{C\}]}\,(\Diamond\langle\,\text{sink}\,\rangle) \;=\; \mathit{ExpRew}^{\mathcal{M}_{\sigma}^{f}\,[\texttt{while}\,(B)\,\{C\}]}\,(\Diamond\langle\,\text{sink}\,\rangle)\,.$$

Lemma 12. *Let $C \in \mathbb{P}$, $f \in \mathbb{E}$, and $\sigma \in \mathbb{S}_{\tau}$. Then*

$$\mathit{ExpRew}^{\mathcal{M}_{\sigma}^{f}\,[C]}\,(\Diamond\langle\,\text{sink}\,\rangle) \;=\; \mathsf{rt}\,[C]\,(f)\,(\sigma).$$

Proof. We will make use of a classical substitution lemma stating that $f(\sigma[x/E]\,) = f[x/E]\,(\sigma)$. The proof is by structural induction on the structure of C.

The Effectless Program $C = \texttt{skip}$. The MC $\mathcal{M}_{\sigma}^{f}\,[\texttt{skip}]$ is of the form

$$\rightarrow \underset{0}{\langle\, \texttt{skip}, \sigma \,\rangle} \rightarrow \underset{f(\sigma[\tau/\tau+1]\,)}{\langle\, \downarrow, \sigma[\tau/\tau+1]\,\rangle} \rightarrow \underset{0}{\langle\, \text{sink} \,\rangle} \circlearrowleft$$

Thus, we have $\Diamond\langle\,\text{sink}\,\rangle = \{\hat{\pi}\}$, where

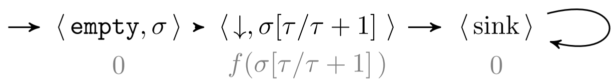
$$\hat{\pi} = \langle\, C, \sigma \,\rangle \langle\, \downarrow, \sigma[\tau/\tau+1]\,\rangle \langle\, \text{sink} \,\rangle.$$

Moreover, $\Pr^{\mathcal{M}_{\sigma}^{f}\,[C]}\{\hat{\pi}\} = 1$ and $rew(\hat{\pi}) = f(\sigma[\tau/\tau+1]\,)$. Thus

$$\mathsf{ExpRew}^{\mathcal{M}_{\sigma}^{f}\,[C]}\,(\Diamond\langle\,\text{sink}\,\rangle)$$

$$
\begin{aligned}
&= \sum_{\pi \in \Diamond\langle\,\mathrm{sink}\,\rangle} \Pr{}^{\mathcal{M}_\sigma^f[C]}\{\pi\} \cdot rew(\pi) \\
&= \Pr{}^{\mathcal{M}_\sigma^f[C]}\{\hat{\pi}\} \cdot rew(\hat{\pi}) \\
&= 1 \cdot f(\sigma[\tau/\tau+1]\,) \\
&= f(\sigma[\tau/\tau+1]\,) \\
&= f[\tau/\tau+1]\,(\sigma) \\
&= \mathsf{rt}\,[C]\,(\sigma)\,.
\end{aligned}
$$

The Empty Program $C = \texttt{empty}$. The MC $\mathcal{M}_\sigma^f\,[\texttt{empty}]$ is of the form



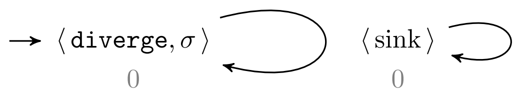
Thus, we have $\Diamond\langle\,\mathrm{sink}\,\rangle = \{\hat{\pi}\}$, where

$$\hat{\pi} = \langle\, C, \sigma \,\rangle\langle\, \downarrow, \sigma \,\rangle\langle\, \mathrm{sink} \,\rangle.$$

Moreover, $\Pr^{\mathcal{M}_\sigma^f[C]}\{\hat{\pi}\} = 1$ and $rew(\hat{\pi}) = f(\sigma)$. Thus

$$
\begin{aligned}
&\mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C]}\,(\Diamond\langle\,\mathrm{sink}\,\rangle) \\
&= \sum_{\pi \in \Diamond\langle\,\mathrm{sink}\,\rangle} \Pr{}^{\mathcal{M}_\sigma^f[C]}\{\pi\} \cdot rew(\pi) \\
&= \Pr{}^{\mathcal{M}_\sigma^f[C]}\{\hat{\pi}\} \cdot rew(\hat{\pi}) \\
&= 1 \cdot f(\sigma) \\
&= f(\sigma) \\
&= \mathsf{rt}\,[C]\,(\sigma)\,.
\end{aligned}
$$

The Diverging Program $C = \texttt{diverge}$. The MC $\mathcal{M}_\sigma^f\,[\texttt{diverge}]$ is of the form



Then $\Diamond\langle\,\mathrm{sink}\,\rangle = \emptyset$ and thus the probability of reaching $\langle\,\mathrm{sink}\,\rangle$ is 0. Thus, we immediately obtain

$$
\begin{aligned}
&\mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C]}\,(\Diamond\langle\,\mathrm{sink}\,\rangle) \\
&= \infty \\
&= \mathsf{rt}\,[C]\,(\sigma)\,.
\end{aligned}
$$

The Erroneous Program $C = \texttt{halt}$. The MC $\mathcal{M}_\sigma^f\,[\texttt{halt}]$ is of the form

$$\begin{array}{ccccc} \rightarrow \langle \texttt{halt}, \sigma \rangle & \longrightarrow & \langle \text{sink} \rangle \circlearrowleft \\ 0 & & 0 \end{array}$$

Thus, each path $\pi \in \Diamond\langle \text{sink} \rangle$ is of the form

$$\hat{\pi} = \langle \texttt{halt}, \sigma \rangle \langle \text{sink} \rangle$$

Moreover, $\Pr^{\mathcal{M}_\sigma^f[C]}\{\pi\} = 1$ and $rew(\pi) = 0$. Then

$$\begin{aligned} & \mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C]}\,(\Diamond\langle \text{sink} \rangle) \\ = & \sum_{\pi \in \Diamond\langle \text{sink} \rangle} \Pr^{\mathcal{M}_\sigma^f[C]}\{\pi\} \cdot rew(\pi) \\ = & \Pr^{\mathcal{M}_\sigma^f[C]}\{\hat{\pi}\} \cdot rew(\hat{\pi}) \\ = & 1 \cdot 0 \\ = & 0 \\ = & \mathsf{rt}\,[C]\,(\sigma)\,. \end{aligned}$$

The Assignment $C = x \coloneqq E$. The MC $\mathcal{M}_\sigma^f\,[x \coloneqq E]$ is of the form

$$\begin{array}{ccccc} \rightarrow \langle x \coloneqq E, \sigma \rangle & \longrightarrow & \langle \downarrow, \sigma[x/E, \tau/\tau+1] \rangle & \longrightarrow & \langle \text{sink} \rangle \circlearrowleft \\ 0 & & f(\sigma[x/E, \tau/\tau+1]) & & 0 \end{array}$$

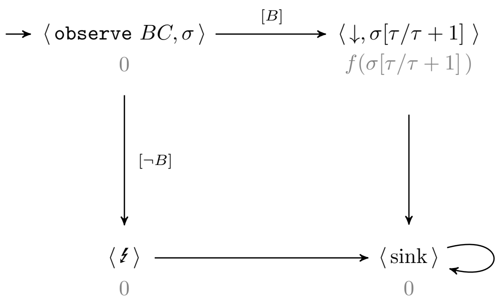
Thus, we have $\Diamond\langle \text{sink} \rangle = \{\hat{\pi}\}$, where

$$\hat{\pi} = \langle C, \sigma \rangle \langle \downarrow, \sigma[x/E, \tau/\tau+1] \rangle \langle \text{sink} \rangle.$$

Moreover, $\Pr^{\mathcal{M}_\sigma^f[C]}\{\hat{\pi}\} = 1$ and $rew(\hat{\pi}) = f(\sigma[x/E, \tau/\tau+1])$. Thus

$$\begin{aligned} & \mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C]}\,(\Diamond\langle \text{sink} \rangle) \\ = & \sum_{\pi \in \Diamond\langle \text{sink} \rangle} \Pr^{\mathcal{M}_\sigma^f[C]}\{\pi\} \cdot rew(\pi) \\ = & \Pr^{\mathcal{M}_\sigma^f[C]}\{\hat{\pi}\} \cdot rew(\hat{\pi}) \\ = & 1 \cdot f(\sigma[\tau/\tau+1]\,) \\ = & f(\sigma[x/E, \tau/\tau+1]) \\ = & f[x/E](\sigma[\tau/\tau+1]) \\ = & f[x/E, \tau/\tau+1](\sigma) \\ = & \mathsf{rt}\,[C]\,(\sigma)\,. \end{aligned}$$

The Observation $C = \texttt{observe}\ B$. The MC $\mathcal{M}_\sigma^f\,[\texttt{observe}\ B]$ is of the form



Two cases arise. If $[B](\sigma) = 1$ we have $\Diamond\langle\,\mathrm{sink}\,\rangle = \{\hat{\pi}\}$, where

$$\hat{\pi} = \langle\, C, \sigma\,\rangle\langle\, \downarrow, \sigma[\tau/\tau+1]\,\rangle\langle\,\mathrm{sink}\,\rangle.$$

Moreover, $\Pr^{\mathcal{M}_\sigma^f[C]}\{\hat{\pi}\} = 1$ and $rew(\hat{\pi}) = f(\sigma[\tau/\tau+1]\,)$. Thus

$$\begin{aligned}
&\mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C]}\left(\Diamond\langle\,\mathrm{sink}\,\rangle\right) \\
=\ & \sum_{\pi\in\Diamond\langle\,\mathrm{sink}\,\rangle} \Pr^{\mathcal{M}_\sigma^f[C]}\{\pi\}\cdot rew(\pi) \\
=\ & \Pr^{\mathcal{M}_\sigma^f[C]}\{\hat{\pi}\}\cdot rew(\hat{\pi}) \\
=\ & 1\cdot f(\sigma[\tau/\tau+1]\,) \\
=\ & f[\tau/\tau+1]\,(\sigma) \\
=\ & [B](\sigma)\cdot f[\tau/\tau+1]\,(\sigma) && ([B](\sigma)=1) \\
=\ & \mathsf{rt}\,[C]\,(\sigma)\,.
\end{aligned}$$

Conversely, if $[B](\sigma) = 0$ we have $\Diamond\langle\,\mathrm{sink}\,\rangle = \{\hat{\pi}\}$, where

$$\hat{\pi} = \langle\, C, \sigma\,\rangle\langle\, \lightning\,\rangle\langle\,\mathrm{sink}\,\rangle.$$

Moreover, $\Pr^{\mathcal{M}_\sigma^f[C]}\{\hat{\pi}\} = 1$ and $rew(\hat{\pi}) = 0$. Thus

$$\begin{aligned}
&\mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C]}\left(\Diamond\langle\,\mathrm{sink}\,\rangle\right) \\
=\ & \sum_{\pi\in\Diamond\langle\,\mathrm{sink}\,\rangle} \Pr^{\mathcal{M}_\sigma^f[C]}\{\pi\}\cdot rew(\pi) \\
=\ & \Pr^{\mathcal{M}_\sigma^f[C]}\{\hat{\pi}\}\cdot rew(\hat{\pi}) \\
=\ & 1\cdot 0 \\
=\ & 0 \\
=\ & [B](\sigma)\cdot f[\tau/\tau+1]\,(\sigma) && ([B](\sigma)=0) \\
=\ & \mathsf{rt}\,[C]\,(\sigma)\,.
\end{aligned}$$

The Sequential Composition $C = C_1 ;\ C_2$.

$$
\begin{aligned}
& \mathsf{ExpRew}^{\mathcal{M}^f_\sigma[C]}(\Diamond\langle\,\mathrm{sink}\,\rangle) \\
=\ & \mathsf{ExpRew}^{\mathcal{M}^{g(C_2,f)}_\sigma[C_1]}(\Diamond\langle\,\mathrm{sink}\,\rangle) && \text{(Lemma 6)} \\
=\ & \mathsf{ExpRew}^{\mathcal{M}^{\mathsf{ExpRew}^{\lambda\rho\,.\,\mathcal{M}^f_\rho[C_2]}(\Diamond\langle\,\mathrm{sink}\,\rangle)}_\sigma[C_1]}(\Diamond\langle\,\mathrm{sink}\,\rangle) \\
=\ & \mathsf{ExpRew}^{\mathcal{M}^{\lambda\rho\,.\,\mathsf{rt}[C_2](f)(\rho)}_\sigma[C_1]}(\Diamond\langle\,\mathrm{sink}\,\rangle) && \text{(I.H. on } C_2\text{)} \\
=\ & \mathsf{ExpRew}^{\mathcal{M}^{\mathsf{rt}[C_2](f)}_\sigma[C_1]}(\Diamond\langle\,\mathrm{sink}\,\rangle) \\
=\ & \mathsf{rt}\,[C_1]\,(\mathsf{rt}\,[C_2]\,(f))\,(\sigma) && \text{(I.H. on } C_1\text{)} \\
=\ & \mathsf{rt}\,[C_1 ;\ C_2]\,(f)\,(\sigma).
\end{aligned}
$$

The Conditional $C = \texttt{if}\ (B)\ \{C_1\}\ \texttt{else}\ \{C_2\}$.

$$
\begin{aligned}
& \mathsf{ExpRew}^{\mathcal{M}^f_\sigma[C]}(\Diamond\langle\,\mathrm{sink}\,\rangle) \\
=\ & [B](\sigma)\cdot\mathsf{ExpRew}^{\mathcal{M}^f_{\sigma[\tau/\tau+1]}[C_1]}(\Diamond\langle\,\mathrm{sink}\,\rangle) && \text{(Lemma 8)} \\
& + [\neg B](\sigma)\cdot\mathsf{ExpRew}^{\mathcal{M}^f_{\sigma[\tau/\tau+1]}[C_2]}(\Diamond\langle\,\mathrm{sink}\,\rangle) \\
=\ & [B](\sigma)\cdot\mathsf{rt}\,[C_1]\,(f)\,(\sigma[\tau/\tau+1]\,) \\
& + [\neg B](\sigma)\cdot\mathsf{rt}\,[C_2]\,(f)\,(\sigma[\tau/\tau+1]\,) && \text{(I.H.)} \\
=\ & ([B](\sigma)\cdot\mathsf{rt}\,[C_1]\,(f)\,(\sigma))\,[\tau/\tau+1] && (B \text{ is } \tau\text{–free}) \\
& + ([\neg B](\sigma)\cdot\mathsf{rt}\,[C_2]\,(f)\,(\sigma))\,[\tau/\tau+1] \\
=\ & ([B](\sigma)\cdot\mathsf{rt}\,[C_1]\,(f)\,(\sigma) + [\neg B](\sigma)\cdot\mathsf{rt}\,[C_2]\,(f)\,(\sigma))\,[\tau/\tau+1] \\
=\ & \mathsf{rt}\,[C]\,(\sigma)\,.
\end{aligned}
$$

The Probabilistic Choice $C = \{C_1\}\ [p]\ \{C_2\}$. Each path $\pi \in \Diamond\langle\,\mathrm{sink}\,\rangle$ either reaches $\langle\, C_1, \sigma[\tau/\tau+1]\,\rangle$ with probability p, or reaches $\langle\, C_2, \sigma[\tau/\tau+1]\,\rangle$ with probability $1-p$. These states are the initial states of the MCs $\mathcal{M}^f_{\sigma[\tau/\tau+1]}[C_1]$ and $\mathcal{M}^f_{\sigma[\tau/\tau+1]}[C_2]$, respectively. Hence,

$$
\begin{aligned}
& \mathsf{ExpRew}^{\mathcal{M}^f_\sigma[C]}(\Diamond\langle\,\mathrm{sink}\,\rangle) \\
=\ & p\cdot\mathsf{ExpRew}^{\mathcal{M}^f_{\sigma[\tau/\tau+1]}[C_1]}(\Diamond\langle\,\mathrm{sink}\,\rangle) + (1-p)\cdot\mathsf{ExpRew}^{\mathcal{M}^f_{\sigma[\tau/\tau+1]}[C_2]}(\Diamond\langle\,\mathrm{sink}\,\rangle) \\
=\ & p\cdot\mathsf{rt}\,[C_1]\,(f)\,(\sigma[\tau/\tau+1]\,) + (1-p)\cdot\mathsf{rt}\,[C_2]\,(f)\,(\sigma[\tau/\tau+1]\,) && \text{(I.H.)} \\
=\ & p\cdot\mathsf{rt}\,[C_1]\,(f)\,(\sigma)[\tau/\tau+1] + (1-p)\cdot\mathsf{rt}\,[C_2]\,(f)\,(\sigma)[\tau/\tau+1] \\
=\ & (p\cdot\mathsf{rt}\,[C_1]\,(f)\,(\sigma) + (1-p)\cdot\mathsf{rt}\,[C_2]\,(f)\,(\sigma))\,[\tau/\tau+1] \\
=\ & \mathsf{rt}\,[C]\,(\sigma)\,.
\end{aligned}
$$

The Loop $C = \texttt{while}\ (B)\ \{C'\}$. For each natural number $k \geq 1$ and $\sigma \in \mathbb{S}_\tau$, we have

$$
\mathsf{rt}\left[\texttt{while}^{<k}\ (B)\ \{C'\}\right](f)\,(\sigma)
$$

$$
\begin{aligned}
&= \mathsf{rt}\left[\mathtt{if}\ (B)\ \left\{C';\ \mathtt{while}^{<k}\ (B)\ \{C'\}\right\}\ \mathtt{else}\ \{\mathtt{empty}\}\right](f)(\sigma) \\
&= ([B](\sigma)\cdot \mathsf{rt}\left[C';\ \mathtt{while}^{<k}\ (B)\ \{C'\}\right](f)(\sigma) \\
&\quad + [\neg B](\sigma)\cdot \mathsf{rt}\left[\mathtt{empty}\right](f)(\sigma))[\tau/\tau+1] \\
&= ([B](\sigma)\cdot \mathsf{rt}\left[C';\ \mathtt{while}^{<k}\ (B)\ \{C'\}\right](f)(\sigma[\tau/\tau+1]) \\
&\quad + [\neg B](\sigma)\cdot \mathsf{rt}\left[\mathtt{empty}\right](f)(\sigma[\tau/\tau+1])) \\
&= ([B](\sigma)\cdot \mathsf{ExpRew}^{\mathcal{M}^f_{\sigma[\tau/\tau+1]}\left[\mathtt{while}^{<k}\,(B)\,\{C'\}\right]}(\Diamond\langle\,\mathrm{sink}\,\rangle) && \text{(I.H.)} \\
&\quad + [\neg B](\sigma)\cdot \mathsf{ExpRew}^{\mathcal{M}^f_{\sigma[\tau/\tau+1]}[\mathtt{empty}]}(\Diamond\langle\,\mathrm{sink}\,\rangle)) \\
&= \mathsf{ExpRew}^{\mathcal{M}^f_{\sigma}\left[\mathtt{while}^{<k}\,(B)\,\{C'\}\right]}(\Diamond\langle\,\mathrm{sink}\,\rangle)\,.
\end{aligned}
$$

Together with the already proven equation

$$
\mathsf{rt}\left[\mathtt{halt}\right](f)(\sigma) \;=\; \mathsf{ExpRew}^{\mathcal{M}^f_{\sigma}[\mathtt{halt}]}(\Diamond\langle\,\mathrm{sink}\,\rangle)
$$

we conclude

$$
\begin{aligned}
&\mathsf{rt}\left[\mathtt{while}\ (B)\ \{C'\}\right](f)(\sigma) \\
&= \sup_{k\in\mathbb{N}} \mathsf{rt}\left[\mathtt{while}^{<k}\ (B)\ \{C'\}\right](f)(\sigma) && \text{(Lemma 7)} \\
&= \sup_{k\in\mathbb{N}} \mathsf{ExpRew}^{\mathcal{M}^f_{\sigma}\left[\mathtt{while}^{<k}\,(B)\,\{C'\}\right]}(\Diamond\langle\,\mathrm{sink}\,\rangle) \\
&= \mathsf{ExpRew}^{\mathcal{M}^f_{\sigma}\left[\mathtt{while}\,(B)\,\{C'\}\right]}(\Diamond\langle\,\mathrm{sink}\,\rangle)\,. && \text{(Lemma 11)}
\end{aligned}
$$

Lemma 13. *Let $C \in \mathbb{P}$, $f \in \mathbb{E}$, and $\sigma \in \mathbb{S}_\tau$. Then*

$$
\mathit{ExpRew}^{\mathcal{M}^f_{\sigma}[C]}(\Diamond\langle\,\mathrm{sink}\,\rangle \cap \neg\Diamond\langle\,\lightning\,\rangle) \;=\; \mathit{ExpRew}^{\mathcal{M}^f_{\sigma}[C]}(\Diamond\langle\,\mathrm{sink}\,\rangle)\,.
$$

Proof. Two cases arise. First, assume that the probability of reaching $\langle\,\mathrm{sink}\,\rangle$ in $\mathcal{M}^f_\sigma[C]$ is 1. Since no path $\pi \in \Diamond\langle\,\mathrm{sink}\,\rangle \cap \Diamond\langle\,\lightning\,\rangle$ contains a state of the form $\langle\,\downarrow, \sigma\,\rangle$, we know by Definition 9 that $\mathit{rew}(\pi) = 0$. Hence,

$$
\mathsf{ExpRew}^{\mathcal{M}^f_{\sigma}[C]}(\Diamond\langle\,\mathrm{sink}\,\rangle \cap \Diamond\langle\,\lightning\,\rangle) = 0. \qquad (\bigstar)
$$

Then

$$
\begin{aligned}
&\mathsf{ExpRew}^{\mathcal{M}^f_{\sigma}[C]}(\Diamond\langle\,\mathrm{sink}\,\rangle) \\
&= \sum_{\pi\in\Diamond\langle\,\mathrm{sink}\,\rangle} \mathrm{Pr}^{\mathcal{M}^f_{\sigma}[C]}\{\pi\}\cdot \mathit{rew}(\pi) \\
&= \sum_{\pi\in\Diamond\langle\,\mathrm{sink}\,\rangle\cap\Diamond\langle\,\lightning\,\rangle} \mathrm{Pr}^{\mathcal{M}^f_{\sigma}[C]}\{\pi\}\cdot \mathit{rew}(\pi) \\
&\quad + \sum_{\pi\in\Diamond\langle\,\mathrm{sink}\,\rangle\cap\neg\Diamond\langle\,\lightning\,\rangle} \mathrm{Pr}^{\mathcal{M}^f_{\sigma}[C]}\{\pi\}\cdot \mathit{rew}(\pi)
\end{aligned}
$$

$$= \mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C]}(\Diamond\langle\,\text{sink}\,\rangle \cap \Diamond\langle\,\lightning\,\rangle) + \mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C]}(\Diamond\langle\,\text{sink}\,\rangle \cap \neg\Diamond\langle\,\lightning\,\rangle)$$
$$= \mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C]}(\Diamond\langle\,\text{sink}\,\rangle \cap \neg\Diamond\langle\,\lightning\,\rangle)\,. \qquad \text{(by } (\bigstar))$$

For the second case assume the probability of reaching $\langle\,\text{sink}\,\rangle$ in $\mathcal{M}_\sigma^f[C]$ is less than 1. Then $\Diamond\langle\,\text{sink}\,\rangle \cap \neg\Diamond\langle\,\lightning\,\rangle \subseteq \Diamond\langle\,\text{sink}\,\rangle$ implies

$$\sum_{\pi \in \Diamond\langle\,\text{sink}\,\rangle \cap \neg\Diamond\langle\,\lightning\,\rangle} \Pr^{\mathcal{M}_\sigma^f[C]}\{\pi\} \;\leq\; \sum_{\pi \in \Diamond\langle\,\text{sink}\,\rangle} \Pr^{\mathcal{M}_\sigma^f[C]}\{\pi\} < 1.$$

Thus

$$\mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C]}(\Diamond\langle\,\text{sink}\,\rangle \cap \neg\Diamond\langle\,\lightning\,\rangle) \;=\; \infty \;=\; \mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C]}(\Diamond\langle\,\text{sink}\,\rangle)\,.$$

Lemma 14. *Let $C \in \mathbb{P}$, $f \in \mathbb{E}$, and $\sigma \in \mathbb{S}_\tau$. Moreover, let $0 \leq f(\sigma') \leq 1$ for each $\sigma' \in \mathbb{S}_\tau$. Then*

$$\mathsf{wlp}\,[C]\,(f)\,(\sigma) \;=\; \mathit{ExpRew}^{\mathcal{M}_\sigma^f[C]}(\Diamond\langle\,\text{sink}\,\rangle) + \Pr^{\mathcal{M}_\sigma^f[C]}\{\neg\Diamond\text{sink}\}.$$

Proof. The proof is by structural induction on the syntactic structure of $\mathbb{P}$ programs and works analogously to the proof of Lemma 12. For a detailed proof, we refer to [6].

Lemma 15. *Let $C \in \mathbb{P}$, $f \in \mathbb{E}$, and $\sigma \in \mathbb{S}_\tau$. Then*

$$\mathsf{wlp}\,[C]\,(1)\,(\sigma) \;=\; \Pr^{\mathcal{M}_\sigma^f[C]}\{\neg\Diamond\lightning\}.$$

Proof. We first observe that each state not visiting $\langle\,\lightning\,\rangle$ either

1. visits a state of the form $\langle\,\downarrow, \sigma'\,\rangle$ for some $\sigma' \in \mathbb{S}_\tau$ and then immediately reaches $\langle\,\text{sink}\,\rangle$, or
2. visits a state of the form $\langle\,\texttt{halt}, \sigma'\,\rangle$ for some $\sigma' \in \mathbb{S}_\tau$ and then immediately reaches $\langle\,\text{sink}\,\rangle$, or
3. diverges and never reaches $\langle\,\text{sink}\,\rangle$.

Thus,

$$\begin{aligned}
&\Pr^{\mathcal{M}_\sigma^f[C]}\{\neg\Diamond\lightning\}\\
=\;&\Pr^{\mathcal{M}_\sigma^f[C]}\{\Diamond\{\langle\,\downarrow, \sigma'\,\rangle \mid \sigma' \in \mathbb{S}_\tau\}\} + \Pr^{\mathcal{M}_\sigma^f[C]}\{\Diamond\{\langle\,\texttt{halt}, \sigma'\,\rangle \mid \sigma' \in \mathbb{S}_\tau\}\}\\
&+ \Pr^{\mathcal{M}_\sigma^f[C]}\{\neg\Diamond\text{sink}\}\\
=\;&\Pr^{\mathcal{M}_\sigma^1[C]}\{\Diamond\{\langle\,\downarrow, \sigma'\,\rangle \mid \sigma' \in \mathbb{S}_\tau\}\} + \Pr^{\mathcal{M}_\sigma^1[C]}\{\Diamond\{\langle\,\texttt{halt}, \sigma'\,\rangle \mid \sigma' \in \mathbb{S}_\tau\}\}\\
&+ \Pr^{\mathcal{M}_\sigma^1[C]}\{\neg\Diamond\text{sink}\}\\
=\;&\mathsf{ExpRew}^{\mathcal{M}_\sigma^1[C]}(\Diamond\langle\,\text{sink}\,\rangle) + \Pr^{\mathcal{M}_\sigma^1[C]}\{\neg\Diamond\text{sink}\} && \text{(Lemma 13 using } (\bigstar))\\
=\;&\mathsf{wlp}\,[C]\,(1)\,(\sigma) && \text{(Lemma 14)}
\end{aligned}$$

Theorem 6. *Let $C \in \mathbb{P}$, $f \in \mathbb{E}$, and $\sigma \in \mathbb{S}_\tau$. Then*

$$\mathsf{CExpRew}^{\mathcal{M}_\sigma^f[C]}\left(\Diamond\langle\,\text{sink}\,\rangle \mid \neg\Diamond\langle\,\lightning\,\rangle\right) \;=\; \frac{\mathsf{rt}\,[C]\,(f)\,(\sigma)}{\mathsf{wlp}\,[C]\,(1)\,(\sigma)}.$$

Proof.

$$\begin{aligned}
&\mathsf{CExpRew}^{\mathcal{M}_\sigma^f[C]}\left(\Diamond\langle\,\text{sink}\,\rangle \mid \neg\Diamond\langle\,\lightning\,\rangle\right)\\
=\;& \frac{\mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C]}\left(\Diamond\langle\,\text{sink}\,\rangle \cap \neg\Diamond\langle\,\lightning\,\rangle\right)}{\Pr^{\mathcal{M}_\sigma^f[C]}\{\neg\Diamond\langle\,\lightning\,\rangle\}} && (\text{Def. } \mathsf{CExpRew})\\
=\;& \frac{\mathsf{ExpRew}^{\mathcal{M}_\sigma^f[C]}\left(\Diamond\langle\,\text{sink}\,\rangle\right)}{\Pr^{\mathcal{M}_\sigma^f[C]}\{\neg\Diamond\langle\,\lightning\,\rangle\}} && (\text{Lemma 13})\\
=\;& \frac{\mathsf{rt}\,[C]\,(f)\,(\sigma)}{\Pr^{\mathcal{M}_\sigma^f[C]}\{\neg\Diamond\langle\,\lightning\,\rangle\}} && (\text{Lemma 12})\\
=\;& \frac{\mathsf{rt}\,[C]\,(f)\,(\sigma)}{\mathsf{wlp}\,[C]\,(1)\,(\sigma)}. && (\text{Lemma 15})
\end{aligned}$$

A.8 Continuity of rt

Lemma 16. *For each program $C \in \mathbb{P}$ and every ω–chain $f_1 \preceq f_2 \preceq \ldots$, we have* $\mathsf{rt}\,[C]\,(\sup_n f_n) \;=\; \sup_n \mathsf{rt}\,[C]\,(f_n)$.

Proof. Let $f_1 \preceq f_2 \preceq \ldots$ be an ω–chain. We show

$$\mathsf{rt}\,[C]\left(\sup_n f_n\right) = \sup_n \mathsf{rt}\,[C]\,(f_n)$$

by induction on the structure of C. The makes of of the well known fact that continuous functions are closed under substitution[9].

The Effectless Program $C = \texttt{skip}$

$$\begin{aligned}
&\mathsf{rt}\,[\texttt{skip}]\left(\sup_n f_n\right)\\
=\;& \left(\sup_n f_n\right)[\tau/\tau+1] && (\text{Def. } \mathsf{rt})\\
=\;& \sup_n\left(f_n[\tau/\tau+1]\right)\\
=\;& \sup_n \mathsf{rt}\,[\texttt{skip}]\,(f_n)\,.
\end{aligned}$$

[9] cf. Theorem 1.3 in D. Scott. "Data types as lattices." SIAM Journal on computing 5.3 (1976): 522-587.

The Empty Program $C = \texttt{empty}$

$$\begin{aligned} & \mathsf{rt}\,[\texttt{empty}]\left(\sup_n f_n\right) \\ =\ & \sup_n f_n && \text{(Def. } \mathsf{rt}\text{)} \\ =\ & \sup_n \mathsf{rt}\,[\texttt{empty}]\,(f_n)\,. && \text{(Def. } \mathsf{rt}\text{)} \end{aligned}$$

The Assignment $C = x \coloneqq E$

$$\begin{aligned} & \mathsf{rt}\,[x \coloneqq E]\left(\sup_n f_n\right) \\ =\ & (\sup_n f_n)[x/E, \tau/\tau+1] && \text{(Def. } \mathsf{rt}\text{)} \\ =\ & \left(\sup_n f_n[x/E]\right)[\tau/\tau+1] \\ =\ & \sup_n \left(f_n[x/E, \tau/\tau+1]\right) \\ =\ & \sup_n \mathsf{rt}\,[x \coloneqq E]\,(f_n) && \text{(Def. } \mathsf{rt}\text{)} \end{aligned}$$

The Diverging Program $C = \texttt{diverge}$

$$\begin{aligned} & \mathsf{rt}\,[\texttt{diverge}]\left(\sup_n f_n\right) \\ =\ & \boldsymbol{\infty} && \text{(Def. } \mathsf{rt}\text{)} \\ =\ & \sup_n \boldsymbol{\infty} \\ =\ & \sup_n \mathsf{rt}\,[\texttt{diverge}]\,(f_n) && \text{(Def. } \mathsf{rt}\text{)} \end{aligned}$$

The Erroneous Program $C = \texttt{halt}$

$$\begin{aligned} & \mathsf{rt}\,[\texttt{halt}]\left(\sup_n f_n\right) \\ =\ & \mathbf{0} && \text{(Def. } \mathsf{rt}\text{)} \\ =\ & \sup_n \mathbf{0} \\ =\ & \sup_n \mathsf{rt}\,[\texttt{halt}]\,(f_n) && \text{(Def. } \mathsf{rt}\text{)} \end{aligned}$$

The Observation $C = \texttt{observe}\ B$

$$\begin{aligned} & \mathsf{rt}\,[\texttt{observe}\ B]\left(\sup_n f_n\right) \\ =\ & [B] \cdot \left(\sup_n f_n\right)[\tau/\tau+1] && \text{(Def. } \mathsf{rt}\text{)} \end{aligned}$$

$$
\begin{aligned}
&= [B] \cdot \left(\sup_n f_n[\tau/\tau+1] \right) \\
&= \sup_n \left([B] \cdot f_n[\tau/\tau+1] \right) \\
&= \sup_n \mathsf{rt}\,[\texttt{observe}~B]\,(f_n)\,. && \text{(Def. } \mathsf{rt}\text{)}
\end{aligned}
$$

The Sequential Composition $C = C_1;\ C_2$

$$
\begin{aligned}
&\mathsf{rt}\,[C_1;\ C_2]\left(\sup_n f_n\right) \\
&= \mathsf{rt}\,[C_1]\left(\mathsf{rt}\,[C_2]\left(\sup_n f_n\right)\right) && \text{(Def. } \mathsf{rt}\text{)} \\
&= \mathsf{rt}\,[C_1]\left(\sup_n \mathsf{rt}\,[C_2]\,(f_n)\right) && \text{(I.H. on } C_2\text{)} \\
&= \sup_n \mathsf{rt}\,[C_1]\,(\mathsf{rt}\,[C_2]\,(f_n)) && \text{(I.H. on } C_1\text{)} \\
&= \sup_n \mathsf{rt}\,[C_1;\ C_2]\,(f_n)\,.
\end{aligned}
$$

The Conditional $C = \texttt{if}~(B)~\{C_1\}~\texttt{else}~\{C_2\}$ The proof relies on a Monotone Sequence Theorem (MCT) which states that if a sequence $f_1 \preceq f_2 \preceq \ldots$ is a monotonic sequence in $\mathbb{R}^{\infty}_{\geq 0}$ then $\sup_n f_n = \lim_{n\to\infty} f_n$.

$$
\begin{aligned}
&\mathsf{rt}\,[\texttt{if}~(B)~\{C_1\}~\texttt{else}~\{C_2\}]\left(\sup_n f_n\right) \\
&= \left([B]\cdot \mathsf{rt}\,[C_1]\left(\sup_n f_n\right) + [\neg B]\cdot \mathsf{rt}\,[C_2]\left(\sup_n f_n\right)\right)[\tau/\tau+1] && \text{(Def. } \mathsf{rt}\text{)} \\
&= [B]\cdot \mathsf{rt}\,[C_1]\left(\sup_n f_n\right)[\tau/\tau+1] + [\neg B]\cdot \mathsf{rt}\,[C_2]\left(\sup_n f_n\right)[\tau/\tau+1] \\
&= [B]\cdot \sup_n \left(\mathsf{rt}\,[C_1]\,(f_n)\,[\tau/\tau+1]\right) + [\neg B]\cdot \sup_n \left(\mathsf{rt}\,[C_2]\,(f_n)\,[\tau/\tau+1]\right) && \text{(I.H.)} \\
&= [B]\cdot \lim_{n\to\infty} \left(\mathsf{rt}\,[C_1]\,(f_n)\,[\tau/\tau+1]\right) + [\neg B]\cdot \lim_{n\to\infty} \left(\mathsf{rt}\,[C_2]\,(f_n)\,[\tau/\tau+1]\right) \\
& && \text{(MCT)} \\
&= \lim_{n\to\infty} \left([B]\cdot \mathsf{rt}\,[C_1]\,(f_n)\,[\tau/\tau+1] + [\neg B]\cdot \mathsf{rt}\,[C_2]\,(f_n)\,[\tau/\tau+1]\right) \\
&= \sup_n \left([B]\cdot \mathsf{rt}\,[C_1]\,(f_n)\,[\tau/\tau+1] + [\neg B]\cdot \mathsf{rt}\,[C_2]\,(f_n)\,[\tau/\tau+1]\right) && \text{(MCT)} \\
&= \sup_n \left([B]\cdot \mathsf{rt}\,[C_1]\,(f_n) + [\neg B]\cdot \mathsf{rt}\,[C_2]\,(f_n)\right)[\tau/\tau+1] \\
&= \sup_n \mathsf{rt}\,[\texttt{if}~(B)~\{C_1\}~\texttt{else}~\{C_2\}]\,(f_n) && \text{(Def. } \mathsf{rt}\text{)}
\end{aligned}
$$

The Probabilistic Choice $C = \{C_1\}~[p]~\{C_2\}$ The proof relies on a Monotone Sequence Theorem (MCT) which states that if a sequence $f_1 \preceq f_2 \preceq \ldots$ is a monotonic sequence in $\mathbb{R}_{\geq 0}^{\infty}$ then $\sup_n f_n = \lim_{n\to\infty} f_n$.

$$
\begin{aligned}
& \mathsf{rt}\left[\{C_1\}\ [p]\ \{C_2\}\right]\left(\sup_n f_n\right) \\
= \quad & \left(p \cdot \mathsf{rt}\left[C_1\right]\left(\sup_n f_n\right) + (1-p)\cdot \mathsf{rt}\left[C_2\right]\left(\sup_n f_n\right)\right)\left[\tau/\tau+1\right] && \text{(Def. rt)} \\
= \quad & p \cdot \mathsf{rt}\left[C_1\right]\left(\sup_n f_n\right)\left[\tau/\tau+1\right] + (1-p)\cdot \mathsf{rt}\left[C_2\right]\left(\sup_n f_n\right)\left[\tau/\tau+1\right] \\
= \quad & p \cdot \sup_n \left(\mathsf{rt}\left[C_1\right](f_n)\left[\tau/\tau+1\right]\right) + (1-p)\cdot \sup_n \left(\mathsf{rt}\left[C_2\right](f_n)\left[\tau/\tau+1\right]\right) && \text{(I.H.)} \\
= \quad & p \cdot \lim_{n\to\infty} \left(\mathsf{rt}\left[C_1\right](f_n)\left[\tau/\tau+1\right]\right) + (1-p)\cdot \lim_{n\to\infty} \left(\mathsf{rt}\left[C_2\right](f_n)\left[\tau/\tau+1\right]\right) \\
& && \text{(MCT)} \\
= \quad & \lim_{n\to\infty} p \cdot \left(\mathsf{rt}\left[C_1\right](f_n)\left[\tau/\tau+1\right]\right) + (1-p)\cdot \left(\mathsf{rt}\left[C_2\right](f_n)\left[\tau/\tau+1\right]\right) \\
= \quad & \sup_n p \cdot \left(\mathsf{rt}\left[C_1\right](f_n)\left[\tau/\tau+1\right]\right) + (1-p)\cdot \left(\mathsf{rt}\left[C_2\right](f_n)\left[\tau/\tau+1\right]\right) && \text{(MCT)} \\
= \quad & \sup_n \left(p \cdot \mathsf{rt}\left[C_1\right](f_n) + (1-p)\cdot \mathsf{rt}\left[C_2\right](f_n)\right) subst\tau\tau+1 \\
= \quad & \sup_n \mathsf{rt}\left[\{C_1\}\ [p]\ \{C_2\}\right](f_n)\,. && \text{(Def. rt)}
\end{aligned}
$$

The Loop $C = \texttt{while}\ (B)\ \{C'\}$ Let

$$F_f(X) \;=\; \left([\neg B]\cdot f + [B]\cdot \mathsf{rt}\left[C'\right](X)\right)\left[\tau/\tau+1\right].$$

We make use of two facts concerning continuous functions. Fact 1 states that $F_{\sup_n f_n}(X) = \sup_n F_{f_n}(X)$ and follows from a straightforward reasoning. Fact 2 states that $\sup_n F_{f_n}$ is continuous, because $F_{f_1} \preceq F_{f_2} \preceq \ldots$ is an ω–chain of continuous transformers (note that by I.H. $\mathsf{rt}\left[C'\right](f)$ is continuous). Then

$$
\begin{aligned}
& \mathsf{rt}\left[\texttt{while}\ (B)\ \{C'\}\right]\left(\sup_n f_n\right) \\
= \quad & \mathsf{lfp}\, X\ .\ \left(F_{\sup_n f_n}(X)\right) && \text{(Def. rt)} \\
= \quad & \sup_{k\in\mathbb{N}} F^k_{\sup_n f_n}(\mathbf{0}) && \text{(Tarski's FP-Theorem)} \\
= \quad & \sup_{k\in\mathbb{N}} \left(\sup_n F^k_{f_n}\right)(\mathbf{0}) && \text{(Fact 1)} \\
= \quad & \sup_{k\in\mathbb{N}} \sup_n F^k_{f_n}(\mathbf{0}) \\
= \quad & \sup_n \left(\sup_{k\in\mathbb{N}} F^k_{f_n}(\mathbf{0})\right) && \text{(Fact 2)} \\
= \quad & \sup_n \mathsf{rt}\left[\texttt{while}\ (B)\ \{C'\}\right](f_n)\,. && \text{(Def. rt)}
\end{aligned}
$$