A Pre-expectation Calculus for Probabilistic Sensitivity

ALEJANDRO AGUIRRE, IMDEA Software Institute, Spain and Universidad Politécnica de Madrid, Spain
GILLES BARTHE, MPI-SP, Germany and IMDEA Software Institute, Spain
JUSTIN HSU, University of Wisconsin–Madison, USA
BENJAMIN LUCIEN KAMINSKI, University College London, UK
JOOST-PIETER KATOEN, RWTH Aachen University, Germany
CHRISTOPH MATHEJA, RWTH Aachen University, Germany and ETH Zurich, Switzerland

Sensitivity properties describe how changes to the input of a program affect the output, typically by upper bounding the distance between the outputs of two runs by a monotone function of the distance between the corresponding inputs. When programs are probabilistic, the distance between outputs is a distance between distributions. The Kantorovich lifting provides a general way of defining a distance between distributions by lifting the distance of the underlying sample space; by choosing an appropriate distance on the base space, one can recover other usual probabilistic distances, such as the Total Variation distance. We develop a relational pre-expectation calculus to upper bound the Kantorovich distance between two executions of a probabilistic program. We illustrate our methods by proving algorithmic stability of a machine learning algorithm, convergence of a reinforcement learning algorithm, and fast mixing for card shuffling algorithms. We also consider some extensions: using our calculus to show convergence of Markov chains to the uniform distribution over states and an asynchronous extension to reason about pairs of program executions with different control flow.

CCS Concepts: • **Mathematics of computing** → *Probability and statistics*; • **Theory of computation** → *Logic and verification*; *Program reasoning*.

Additional Key Words and Phrases: probabilistic programming, verification

ACM Reference Format:
Alejandro Aguirre, Gilles Barthe, Justin Hsu, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Christoph Matheja. 2021. A Pre-expectation Calculus for Probabilistic Sensitivity. *Proc. ACM Program. Lang.* 5, POPL, Article 52 (January 2021), 28 pages. https://doi.org/10.1145/3434333

1 INTRODUCTION

Sensitivity properties describe how much a change in a function's input can affect the corresponding output, where the degree of change is quantified by a distance function d_{in} on inputs and a distance function d_{out} on outputs. By varying these distances, sensitivity properties appear across many application areas, including: (i) numerical computations, where functions map from reals to reals, and the distances on inputs and outputs are the usual distance on real numbers; (ii) numerical queries, where functions map from a set of records to a number, and the distance between inputs is the number of differing entries; and (iii) learning algorithms, where functions map from a

Authors' addresses: Alejandro Aguirre, IMDEA Software Institute, Spain, Universidad Politécnica de Madrid, Spain; Gilles Barthe, MPI-SP, Germany, IMDEA Software Institute, Spain; Justin Hsu, University of Wisconsin–Madison, USA; Benjamin Lucien Kaminski, University College London, UK; Joost-Pieter Katoen, RWTH Aachen University, Germany; Christoph Matheja, RWTH Aachen University, Germany, ETH Zurich, Switzerland.

2475-1421/2021/1-ART52
https://doi.org/10.1145/3434333

training set to a learned model and where the distance between two training sets is the number of differing examples, and the distance between outputs measures the difference in errors labeling unseen examples. This paper is concerned with sensitivity properties of probabilistic programs. Since such programs produce *distributions* over their output space, the corresponding notions of sensitivity use distances over distributions. The *Total Variation* (TV) distance (a.k.a. statistical distance), for example, is a widely used notion of distance that measures the maximal difference between probabilities of the same event. One key benefit of the TV distance is that it is defined for distributions over arbitrary spaces. However, it is often useful to consider distances inherited from the underlying space. In this setting, the so-called *Kantorovich* metric gives a generic method to lift a distance $\mathcal{E}$ on a ground set X to a distance $\mathcal{E}^{\#}$ on distributions over X. The class of Kantorovich metrics cover many notions of distance. For instance, the TV distance can be obtained by applying the Kantorovich lifting to the discrete distance, which assigns distance 1 to any pair of distinct points, and distance 0 to any pair of equal points.

Approach. We develop a *relational expectation calculus* for reasoning about sensitivity of probabilistic computations under the Kantorovich metric. Relational expectations are mappings expressing a quantitative relation (e.g., a distance or metric) between states, and are modelled as maps of the form $\mathbf{State}\times\mathbf{State} \to [0, \infty]$. The heart of our system is a *relational expectation pre-expectation transformer*, which takes as input a probabilistic program c written in a core imperative language, and a relational expectation $\mathcal{E}$ between output states, and produces a relational expectation $rpe(c,\ \mathcal{E})$ between input states. The calculus is a sound approximation of sensitivity, in the sense that running the program c on inputs at distance smaller than $rpe(c,\ \mathcal{E})$ yields output distributions at distance smaller than $\mathcal{E}^{\#}$.

Technically, our calculus draws inspiration from early work on probabilistic dynamic logic due to Kozen [1985] in which maps $\mathcal{E}\colon \mathbf{State} \to [0, \infty]$ serve as quantitative counterparts of Boolean predicates $P\colon \mathbf{State} \to \{0, 1\}$. McIver and Morgan [2005] later coined the term *expectation*—not to be confused with expected values—for such maps $\mathcal{E}$. Moreover, they developed a weakest pre-expectation calculus for the probabilistic imperative language pGCL. Their calculus was designed as a generalization of Dijkstra's weakest pre-conditions that supports both probabilistic and non-deterministic choice. The basic idea is to define an operator $wpe(c, \mathcal{E})$ that transforms an expectation $\mathcal{E}$ averaged over the *output* distribution of a program c into an expectation evaluated over the *input* state. In this way, the expectation is transformed by the effects of the probabilistic program in a backwards fashion, much like how predicates are transformed through Dijkstra's weakest pre-conditions. Our pre-expectation calculus operates similarly, but—as it aims to measure distances between distributions of outputs in terms of inputs—manipulates *relational* expectations instead. We next motivate the need for relational expectations, and explain why they are challenging.

Why do we need relational pre-expectations? The classical weakest pre-expectation calculus enjoys strong theoretical properties: in particular, it is both sound and complete (in an extensional and intensional sense [Batz et al. 2021]) w.r.t. common program semantics (cf. Gretz et al. [2014]). Therefore, weakest pre-expectations can—in principle—be applied to reason about bounds on the Total Variation distance: Given a program c, (i) take a copy c' over a fresh set of program variables—e.g. if variable x appears in c, substitute it by x' in c'—and (ii) determine the weakest pre-expectation $wpe(c; c', \mathcal{E})$, where the expectation $\mathcal{E}$ measures the distance between variables in c and their counterparts in c'.

However, this naïve approach is not practical for analyzing sensitivity: the TV distance, for example, is defined as a maximum of a difference of probabilities over all events of the output space—to compute the TV distance, we would need to compute the probability of every single output event. Moreover, the above approach pushes the difficulty of reasoning about sensitivity properties into the task of finding suitable invariants for probabilistic programs—a highly challenging task on its own. In particular, finding invariants may involve reasoning about probabilistic independence, which is not readily available when using weakest pre-expectations. In fact, mathematicians have long observed that reasoning about the TV distance or the Kantorovich metric directly from their definition is often intractable. Rather, they rely various approximations to bound the distance between distributions. One classical method is based on *probabilistic couplings* [Villani 2008], a mathematical tool for relating two different distributions. Relational pre-expectations naturally connect with probabilistic couplings, and capture well-established proof principles used by mathematicians for reasoning about the TV distance.

Challenges of relational pre-expectations. Relational pre-expectations pose a number of specific challenges compared to their unary counterpart. First, the Kantorovich distance cannot be defined inductively on the structure of programs. More specifically, the Kantorovich distance between two runs of $c; c'$ is not a simple combination of the Kantorovich distances between two runs of c and two runs of c' (we provide a counterexample in Section 3). Instead, we define a pre-expectation calculus $\widetilde{rpe}(c, \mathcal{E})$ that gives a compositional *upper-bound* of the Kantorovich distance—this is sufficient for proving sensitivity properties.

Second, the proof of soundness for our relational pre-expectation calculus is significantly more involved than for the usual weakest pre-expectation calculus, and use results from optimal transport theory. In particular, we are only able to prove soundness for discrete distributions.

Third, relational calculi are naturally better suited to reason about two executions that follow the same control-flow. We offer useful support for reasoning about executions with different control-flow, through a careful generalization of the rules for conditionals and loops. While our rules do not suffice for arbitrary examples—it remains an open problem to develop relatively complete verification approaches for relational properties of probabilistic programs—they suffice for non-trivial examples that exhibit asynchronous behavior.

Applications. We demonstrate our technique on several applications. In our first application, we formalize an *algorithmic stability* property of machine-learning algorithms. Informally, algorithmic stability describes how much the parameters produced by a learning algorithm are affected when one input training example is changed; this notion of probabilistic sensitivity is known to imply generalization and prevent overfitting [Bousquet and Elisseeff 2002]. We use our calculus for proving algorithmic stability of a commonly-used learning algorithm: stochastic gradient descent (SGD); while this example has been verified in prior work [Barthe et al. 2018], we show how our approach enables a substantially simpler proof.

Then, we consider a pair of applications showing convergence properties. We first formalize convergence of a reinforcement learning algorithm [Sutton 1988], inspired by a recent analysis by Amortila et al. [2020]. For our most challenging examples, we show convergence and rapid mixing of several card shuffling algorithms [Aldous 1983]. We show that the TV distance between the outputs of two probabilistic loops decreases to 0 as the number of loop iterations increases—that is, the output distributions from any two inputs *converge* to the same distribution. Moreover, our technique is precise enough to prove quantitative bounds on the rate of convergence; upper bounds on convergence speed are key properties when analyzing algorithms that generate samples form complex distributions, such as Markov Chain Monte Carlo.

Extensions: uniformity. Then, we show how to formalize other properties complementing our bounds on convergence rate. Using our system, we demonstrate how to prove that a probabilistic program generates a uniform distribution. This relies on showing that for any two final states s_1, s_2, the absolute difference of the probability of finishing at s_1 and the probability of s_2 goes to 0 as the number of iterations increases.

Extensions: asynchronous reasoning. Finally, we describe extensions to our calculus for asynchronous reasoning, allowing us to prove relational properties when pairs of program executions have different control flow. We demonstrate our asynchronous extensions to reason about a program generating a binomial distribution.

Contributions and outline. After introducing preliminaries on probability theory and the Kantorovich distance (§ 2), we present our main contributions:

- We define a sound, compositional, relational pre-expectation calculus for determining upper-bounds on the Kantorovich distance. We introduce convenient proof rules for sampling commands and loops, and we show that the core fragment of a previous probabilistic relational Hoare logic $\mathbb{E}$pRHL [Barthe et al. 2018] can be embedded into our calculus (§ 3).
- We apply our calculus to three case studies. As a warmup example, we use our calculus to provide a clean proof of algorithmic stability of stochastic gradient descent [Hardt et al. 2016] (§ 4). Second, we formalize convergence of TD(0), an algorithm from the Reinforcement Learning literature [Sutton 1988] (§ 5). Third, we apply our calculus to show rapid convergence of random walks and card shuffling algorithms [Aldous 1983] (§ 6).
- As a complementary extension to the previous examples, we use our calculus to show that the limiting distribution of a card-shuffling routine is uniform (§ 7).
- We present proof rules for reasoning about programs with asynchronous control flow (§ 8).

Finally, we survey related work (§ 9) and conclude (§ 10).

This is the proceedings version of the paper. Omitted proofs and details can be found in the full version: https://arxiv.org/pdf/1901.06540.pdf.

2 MATHEMATICAL PRELIMINARIES

We briefly recap the foundations required for relational reasoning about sensitivity properties: (1) probability theory, (2) probabilistic programming languages, and (3) distances on probability distributions. A comprehensive treatment of these topics is found, e.g., in the textbooks [Ash and Doleans-Dade 2000; McIver and Morgan 2005; Villani 2008].

2.1 Basic Probability Concepts

We will use sub-distributions to model probabilistic behavior. A *sub-distribution* over a countable set A is a function $\mu\colon A \to [0,1]$ assigning a probability to each element of A. Probabilistic *events* are subsets $B \subseteq A$; the probability of B is denoted $\mu(B)$ and defined by $\mu(B) = \sum_{b \in B} \mu(b)$. The *support* of μ is the set of all events $a \in A$ with $\mu(a) > 0$. Moreover, we let $|\mu| = \mu(A)$. As usual, the probabilities in any sub-distribution must sum up to at most 1, i.e. $|\mu| \leq 1$. We call μ a distribution if $|\mu| = 1$. We let $\mathbf{Dist}(A)$ denote the set of *sub-distributions* over A.

Given a sub-distribution $\mu \in \mathbf{Dist}(A_1 \times A_2)$ over a product set, its left and right *marginals*, $\pi_1(\mu)$ and $\pi_2(\mu)$, are sub-distributions over A_1 and A_2, respectively, which are given by $\pi_1(\mu)(x_1) = \sum_{x_2 \in X} \mu(x_1, x_2)$, and $\pi_2(\mu)(x_2) = \sum_{x_1 \in X} \mu(x_1, x_2)$.

The *Dirac distribution* $\delta(a) \in \mathbf{Dist}(A)$ is the point distribution at $a \in A$, $\delta(a)(a') = [a = a']$, where the right-hand-side is an *Iverson-bracket* which evaluates to 1 if the formula inside (in this case, $a = a'$) evaluates to true, and to 0 otherwise. If $f\colon A \to \mathbb{R}^{\infty}_{\geq 0}$ is a function mapping into the extended reals, we can take its *expected value* $\mathbb{E}_\mu[f]$ with respect to some sub-distribution $\mu \in \mathbf{Dist}(A)$: $\mathbb{E}_\mu[f] = \sum_{a \in A} f(a) \cdot \mu(a)$. If the sum diverges, the expected value is defined to be ∞. We assume that addition and multiplication are extended in the natural way, with the convention $0 \cdot \infty = \infty \cdot 0 = 0$.

2.2 Programming Language and Semantics

We work with a standard probabilistic imperative language pWHILE. This language has commands defined by the following grammar:

$$c \coloneqq \mathbf{skip} \mid x \leftarrow e \mid x \overset{\$}{\leftarrow} d \mid c; c \mid \mathbf{if}\ e\ \mathbf{then}\ c\ \mathbf{else}\ c \mid \mathbf{while}\ e\ \mathbf{do}\ c\ .$$

Variables x are drawn from an arbitrary but finite set **Var** of variable names. Expressions e are largely standard, formed from variables and basic operations (e.g., integer addition, boolean conjunction). To handle programs with (static) arrays, we assume expressions include basic array operations for accessing and updating. For instance, when a is an array variable we use the following syntactic sugar:

$$a[e] \triangleq \mathbf{Lookup}(a, e) \quad \textit{(expression)} \qquad \text{and} \qquad a[e] \leftarrow e' \triangleq a \leftarrow \mathbf{Update}(a, e, e') \quad \textit{(command)}$$

The random sampling command $x \overset{\$}{\leftarrow} d$ takes a sample from some primitive distribution d and stores it in x. For simplicity, we assume that primitive distributions do not have free program variables; it is not difficult to remove this limitation. We also assume that primitive distributions can be interpreted as full distributions $[\![d]\!] : \mathbf{Dist}(D)$ over some countable set D, possibly different for different distributions. We will often use the uniform distribution $U(S)$ when S is a finite, non-empty set; for instance, for a positive integer N we will write $[N]$ for the set of integers $\{0, \dots, N-1\}$, so that $x \overset{\$}{\leftarrow} U([N])$ samples each number in $[N]$ with probability $1/N$ and stores it in x. The distributions can also be parameterized by some more complex expression, for instance in $x \overset{\$}{\leftarrow} [y]$ for a program variable y.

Programs in pWHILE transform *states*, which are finite maps $s : \mathbf{Var} \to D$; we write **State** for the set of all states. The semantics of a program c is a map $[\![c]\!] : \mathbf{State} \to \mathbf{Dist}(\mathbf{State})$ assigning a sub-distribution over possible outputs to each input. For example, for the random sampling command, we define

$$([\![x \overset{\$}{\leftarrow} d]\!] s)(s') \triangleq \begin{cases} d(s'(x)) & : s(y) = s'(y) \text{ for all } y \neq x \\ 0 & : \text{otherwise} \end{cases}$$

The semantics of the remaining language constructs is standard. As we only work with discrete primitive distributions and states have finitely many variables, output distributions programs always have countable support.

To express quantitative distance between pairs of states, we use *relational expectations*, which are maps of type $\mathbf{State} \times \mathbf{State} \to \mathbb{R}^{\infty}_{\geq 0}$. The set **Exp** of all relational expectations is equipped with the pointwise order inherited from the order on $\mathbb{R}^{\infty}_{\geq 0}$, i.e., $\mathcal{E} \leq \mathcal{E}'$ if and only if $\mathcal{E}(s_1, s_2) \leq \mathcal{E}'(s_1, s_2)$ for all pairs (s_1, s_2) of states. Since $\mathbb{R}^{\infty}_{\geq 0}$ is a complete lattice and **Exp** has the pointwise order, **Exp** is also a complete lattice: the top and bottom elements are the constant relational expectations ∞ and 0, which send all pairs of states to ∞ and 0 respectively.

For denoting specific relational expectations, we borrow notation from relational Hoare logic [Benton 2004]: we tag variables with $\langle 1 \rangle$ or $\langle 2 \rangle$ to refer to their value in the first or the second state, respectively. For instance, $[x\langle 1 \rangle = x\langle 2 \rangle]$ is a relational expectation encoding the predicate $\lambda \langle s_1, s_2 \rangle.\ [s_1(x) = s_2(x)]$.

2.3 Distances Between Probability Distributions

Sensitivity properties of probabilistic programs are stated in terms of concrete notions of distances between distributions. A standard example is the following:

Definition 2.1 (Total Variation distance). The Total Variation (TV) distance between $\mu_1, \mu_2 \in \mathbf{Dist}(X)$ is defined as: $TV(\mu_1, \mu_2) \triangleq \frac{1}{2} \sum_{x \in X} \left| \mu_1(x) - \mu_2(x) \right|$.

The term "distance" (or "*metric*") is justified as $TV(\mu_1, \mu_2)$ is symmetric, satisfies the triangle inequality, and maps to zero if and only if $\mu_1 = \mu_2$. The normalization factor of $\frac{1}{2}$ ensures that the TV distance is within $[0, 1]$. Roughly speaking, the TV distance measures the largest difference in probabilities of any event between two given distributions.

Note that the TV distance does not require the underlying set X to be equipped with a metric. If X is a metric space, we can define a more complex distance:

Definition 2.2 (Kantorovich distance). Let X be an (extended) metric space with a distance $\mathcal{E}\colon X \times X \to \mathbb{R}^{\infty}_{\geq 0}$. The *Kantorovich distance* is a canonical lifting of $\mathcal{E}$ to a function $\mathcal{E}^{\#}\colon \mathbf{Dist}(X)\times\mathbf{Dist}(X) \to \mathbb{R}^{\infty}_{\geq 0}$ that defines a metric on $\mathbf{Dist}(X)$. This distance is defined as

$$\mathcal{E}^{\#}(\mu_1, \mu_2) = \inf_{\mu \in \Gamma(\mu_1, \mu_2)} \mathbb{E}_{\mu}[\mathcal{E}],$$

where $\Gamma(\mu_1, \mu_2)$ is the set of *probabilistic couplings* of μ_1, μ_2, given by

$$\Gamma(\mu_1, \mu_2) = \{\mu \in \mathbf{Dist}(X \times X) \mid \pi_i(\mu) = \mu_i, \text{ for } i = 1, 2\}.$$

The set $\Gamma(\mu_1, \mu_2)$ is non-empty provided $|\mu_1| = |\mu_2|$. Otherwise, $\Gamma(\mu_1, \mu_2) = \emptyset$ and $\mathcal{E}^{\#}(\mu_1, \mu_2) = \infty$.

The coupling-based definition of the Kantorovich distance is more abstract than other distances between distributions, but its generality turns out to be a strength. First, we can recover the TV distance as a lifting of the discrete metric:

Theorem 2.3 (Total variation and Kantorovich distance). *Let $\mu_1, \mu_2 \in \mathbf{Dist}(X)$ such that $|\mu_1| = |\mu_2| = 1$. If the discrete metric $\mathcal{E}\colon X \times X \to \{0, 1\}$ is given by $\mathcal{E}(x_1, x_2) = [x_1 \neq x_2]$, then $TV\big(\mu_1, \mu_2\big) = \mathcal{E}^{\#}\big(\mu_1, \mu_2\big)$.*

Another advantage of the Kantorovich distance is that it is defined as an infimum. For our goal of proving sensitivity properties, it suffices to compute an upper bound of the distance, which corresponds to determining $\mathbb{E}_{\mu}[\mathcal{E}]$ for some particular witness coupling μ.

Traditionally, the definition of $\mathcal{E}^{\#}$ is restricted to functions $\mathcal{E}$ defining a metric on X. However, the definition of $\mathcal{E}^{\#}$ extends to arbitrary functions $\mathcal{E}$; we abuse terminology and use the term Kantorovich distance also in the more general case. For instance, we can use this more general notion to bound the difference between the expected values of two functions on the outputs of two program runs:

Theorem 2.4 (Absolute expected difference). *Let $\mu_1, \mu_2 \in \mathbf{Dist}(X)$ such that $|\mu_1| = |\mu_2| = 1$, and let $f_1, f_2\colon X \to \mathbb{R}^{\infty}_{\geq 0}$. Let $\mathcal{E}\colon X \times X \to \mathbb{R}^{\infty}_{\geq 0}$ be defined by $\mathcal{E}(x_1, x_2) = |f_1(x_1) - f_2(x_2)|$. Then $\Big|\mathbb{E}_{\mu_1}[f_1] - \mathbb{E}_{\mu_2}[f_2]\Big| \leq \mathcal{E}^{\#}\big(\mu_1, \mu_2\big)$.*

Note that above, the relational expectation $\mathcal{E}$ need not be a metric. We can also obtain bounds on the TV distance when lifting other base distances that assign a minimum, non-zero distance to all pairs of distinct elements.

Theorem 2.5 (Scaled TV distance). *Let $\mu_1, \mu_2 \in \mathbf{Dist}(X)$ with $|\mu_1| = |\mu_2| = 1$, let $\mathcal{E}_{\rho}\colon X \times X \to [0, 1]$, and let $\rho \in \mathbb{R}_{>0}$ be a strictly positive constant with $\mathcal{E}_{\rho}(x_1, x_2) \geq \rho \cdot [x_1 \neq x_2]$. Then, $TV\big(\mu_1, \mu_2\big) \leq \frac{1}{\rho} \cdot \mathcal{E}^{\#}_{\rho}\big(\mu_1, \mu_2\big)$.*

3 BOUNDING EXPECTED SENSITIVITY WITH RELATIONAL PRE-EXPECTATIONS

As we have seen, the Kantorovich distance encompasses many specific distances on distributions. To reason about a general class of sensitivity properties for probabilistic programs, we aim to bound the Kantorovich distance between two output distributions in terms of the distance between two program inputs. In this section, we develop a relational pre-expectation operation to prove such bounds.

3.1 A First (Unsuccessful) Attempt: a Relational Pre-expectation for Exact Bounds

Since we want to reason about the Kantorovich distance lifting of a relational expectation $\mathcal{E}\colon \mathbf{State}\times \mathbf{State} \to \mathbb{R}^{\infty}_{\geq 0}$ between output distributions of a program c, an initial idea is to define a relational pre-expectation operator $rpe(c,\ \mathcal{E})$ coinciding exactly with the Kantorovich distance:

$$rpe(c,\ \mathcal{E})(s_1,\ s_2) = \mathcal{E}^{\#}\Big(\llbracket c \rrbracket s_1,\ \llbracket c \rrbracket s_2\Big)\ ,$$

and then prove bounds of the form $rpe(c,\ \mathcal{E}_{out}) \leq \mathcal{E}_{in}$ in order to bound the Kantorovich distance between outputs by some distance between inputs. While this definition is appealing, it turns out to be inconvenient for formal reasoning because it does not behave well under sequential composition: the expected sequence rule $rpe(c; c',\ \mathcal{E}) = rpe(c,\ rpe(c',\ \mathcal{E}))$ does *not* hold. Roughly, this is because choosing local infima on each step does not necessarily amount to a global infimum. In fact, in some cases the global infimum cannot be realized by local choices.

Example 3.1. The *Bernoulli* distribution $B(p)$ with bias p returns 1 with probability p and 0 with probability $1-p$. Consider the following programs:

$$\begin{aligned} c &= \textbf{if } b \textbf{ then } x \xleftarrow{\$} B(1/2) \textbf{ else } y \xleftarrow{\$} B(1/2) \\ c' &= \textbf{if } b \textbf{ then } y \xleftarrow{\$} B(1/2) \textbf{ else } x \xleftarrow{\$} B(1/2)\ . \end{aligned}$$

Moreover, consider the relational expectation $\mathcal{E} = [x\langle 1\rangle \neq x\langle 2\rangle \vee y\langle 1\rangle \neq y\langle 2\rangle]$. If we fix $b\langle 1\rangle = \mathsf{true}$ and $b\langle 2\rangle = \mathsf{false}$ throughout, then

$$rpe(c',\ \mathcal{E})(s'_1, s'_2) = \inf_{\Gamma(\llbracket y \xleftarrow{\$} B(1/2)\rrbracket s'_1,\, \llbracket x \xleftarrow{\$} B(1/2)\rrbracket s'_2)} \mathbb{E}[\mathcal{E}]\ .$$

To compute the above relational pre-expectation, we first need to understand the possible couplings. The two target marginal distributions are:

$$\mu_1 \triangleq \llbracket y \xleftarrow{\$} B(1/2)\rrbracket s'_1 = \begin{cases} \frac{1}{2} : x \mapsto s'_1(x), y \mapsto 0 \\ \frac{1}{2} : x \mapsto s'_1(x), y \mapsto 1 \end{cases}$$

$$\mu_2 \triangleq \llbracket x \xleftarrow{\$} B(1/2)\rrbracket s'_2 = \begin{cases} \frac{1}{2} : x \mapsto 0, y \mapsto s'_2(y) \\ \frac{1}{2} : x \mapsto 1, y \mapsto s'_2(y)\ . \end{cases}$$

The marginal conditions for couplings (Definition 2.2) imply that any coupling in $\Gamma(\mu_1, \mu_2)$ is of the form

$$\begin{aligned} \mu_\rho(s_1, s_2) = {} & \rho \cdot [s_1(x) = s'_1(x) \wedge s_1(y) = 1] \cdot [s_2(x) = 1 \wedge s_2(y) = s'_2(y)] \\ & + \left(\tfrac{1}{2} - \rho\right) \cdot [s_1(x) = s'_1(x) \wedge s_1(y) = 1] \cdot [s_2(x) = 0 \wedge s_2(y) = s'_2(y)] \\ & + \left(\tfrac{1}{2} - \rho\right) \cdot [s_1(x) = s'_1(x) \wedge s_1(y) = 0] \cdot [s_2(x) = 1 \wedge s_2(y) = s'_2(y)] \\ & + \rho \cdot [s_1(x) = s'_1(x) \wedge s_1(y) = 0] \cdot [s_2(x) = 0 \wedge s_2(y) = s'_2(y)]\ . \end{aligned}$$

for some $0 \leq \rho \leq \frac{1}{2}$ and the previously fixed states s'_1 and s'_2. Hence,

$$\begin{aligned} \mathbb{E}_{\mu_\rho}[\mathcal{E}] = {} & \rho \cdot [s'_1(x) \neq 1 \vee s'_2(y) \neq 1] + \left(\tfrac{1}{2} - \rho\right)[s'_1(x) \neq 0 \vee s'_2(y) \neq 1] \\ & + \left(\tfrac{1}{2} - \rho\right)[s'_1(x) \neq 1 \vee s'_2(y) \neq 0] + \rho \cdot [s'_1(x) \neq 0 \vee s'_2(y) \neq 0]\ . \end{aligned}$$

Since $rpe(c',\ \mathcal{E})$ takes the minimum over all couplings, i.e., the minimum over all $\rho \in [0, \frac{1}{2}]$, by simple computation we get that $rpe(c',\ \mathcal{E})(s'_1, s'_2) = 1/2$, setting $\rho = 1/2$ if $s'_1(x) = s'_2(y)$ and $\rho = 0$ otherwise. Since $s'_1(x), s'_2(y)$ are sampled from $\llbracket c \rrbracket s_1$ and $\llbracket c \rrbracket s_2$, for any way to couple

$$
\begin{aligned}
\widetilde{rpe}(\mathbf{skip}, \mathcal{E}) &\triangleq \mathcal{E} \\
\widetilde{rpe}(x \leftarrow e, \mathcal{E}) &\triangleq \mathcal{E}\{e\langle 1\rangle, e\langle 2\rangle / x\langle 1\rangle, x\langle 2\rangle\} \\
&\triangleq \lambda s_1 s_2 . \mathcal{E}(s_1[x \mapsto e\langle 1\rangle], s_2[x \mapsto e\langle 2\rangle]) \\
\widetilde{rpe}(x \xleftarrow{\$} d, \mathcal{E}) &\triangleq \lambda s_1 s_2 . \, \mathcal{E}^{\#}(\llbracket x \xleftarrow{\$} d \rrbracket s_1, \llbracket x \xleftarrow{\$} d \rrbracket s_2) \text{ , where } \mathcal{E}^{\#}(\mu_1, \mu_2) \triangleq \inf_{\mu \in \Gamma(\mu_1, \mu_2)} \mathbb{E}_{\mu}[\mathcal{E}] \\
\widetilde{rpe}(c; c', \mathcal{E}) &\triangleq \widetilde{rpe}(c, \widetilde{rpe}(c', \mathcal{E})) \\
\widetilde{rpe}(\mathbf{if}\ e\ \mathbf{then}\ c\ \mathbf{else}\ c', \mathcal{E}) &\triangleq [e\langle 1\rangle \wedge e\langle 2\rangle] \cdot \widetilde{rpe}(c, \mathcal{E}) + [\neg e\langle 1\rangle \wedge \neg e\langle 2\rangle] \cdot \widetilde{rpe}(c', \mathcal{E}) + [e\langle 1\rangle \neq e\langle 2\rangle] \cdot \infty \\
\widetilde{rpe}(\mathbf{while}\ e\ \mathbf{do}\ c, \mathcal{E}) &\triangleq \mathsf{lfp} X . \Phi_{\mathcal{E}, c}(X), \\
\text{where } \Phi_{\mathcal{E}, c}(X) &\triangleq [e\langle 1\rangle \wedge e\langle 2\rangle] \cdot \widetilde{rpe}(c, X) + [\neg e\langle 1\rangle \wedge \neg e\langle 2\rangle] \cdot \mathcal{E} + [e\langle 1\rangle \neq e\langle 2\rangle] \cdot \infty
\end{aligned}
$$

Fig. 1. Definition of the relational pre-expectation operator $\widetilde{rpe}(c, \mathcal{E})$.

them $rpe(c,\ rpe(c',\ \mathcal{E}))(s_1, s_2) = \frac{1}{2} > 0$. However, $\llbracket c; c' \rrbracket s_1$ and $\llbracket c; c' \rrbracket s_2$ have the same marginal distributions for (x, y) and thus distance 0. Therefore,

$$0 = rpe(c; c',\ \mathcal{E})(s_1, s_2) < rpe(c,\ rpe(c',\ \mathcal{E}))(s_1, s_2) = \tfrac{1}{2}\ .$$

Fortunately, we generally do not need to compute the exact Kantorovich distance to prove sensitivity properties: an upper bound suffices. Since the Kantorovich distance is an infimum over *all* couplings, we can establish upper bounds by exhibiting a *specific* coupling—of course, the tightness of these upper bounds will depend on the particular coupling we chose. Crucially, couplings *can* be constructed compositionally: a coupling for a sequential composition $c; c'$ can be obtained by combining a coupling for c with a coupling for c'. We leverage this observation into our compositional relational pre-expectation calculus, which provides upper bounds on the Kantorovich distance.

3.2 Compositional Upper Bounds by Relational Pre-expectation

To facilitate compositional reasoning, we define an upper bound $\widetilde{rpe}(c, \mathcal{E})$ of the Kantorovich distance $\mathcal{E}$ with respect to program c. Technically, $\widetilde{rpe}(c, \mathcal{E})$ is a relational pre-expectation calculus defined by induction on the structure of c, similarly to the calculus by Morgan et al. [1996]. The rules of our calculus are shown in Figure 1. We take the indicator expectation $[\mathcal{P}]$ to be 1 if $\mathcal{P}$ is true, otherwise 0, and we define addition and multiplication on expectations pointwise. The cases of skipping, assignments and sequential composition are straightforward and apply the backwards semantics of commands. The relational pre-expectation of sampling is expressed directly in terms of the Kantorovich distance, i.e., an infimum is taken over the set of all couplings, which is not always possible in practice. We give more details on this problem in Section 3.3. The relational pre-expectation for conditionals assumes the two runs are synchronized. If not, $[e\langle 1\rangle \neq e\langle 2\rangle] = 1$ and the distance is (trivially) upper bounded by ∞, since the branches may not terminate with the same probability, so the set of couplings may be empty. Finally, in the case of while loops, we take the least fixed point of the characteristic functional $\Phi_{\mathcal{E}, c}$ of the loop. It is not hard to show that $\Phi_{\mathcal{E}, c}(-)\colon \mathbf{Exp} \to \mathbf{Exp}$ is monotonic, so by the Knaster-Tarski theorem the least fixed point exists. As in the previous case, the relational pre-expectation returns ∞ when runs are not synchronized, i.e., only one loop guard is true. However, if the loop does not termine on *both* sides with probability one, the least fixed point becomes zero, i.e., we measure no distance between two diverging programs. Computing the exact least fixed point is usually not possible. We present an invariant-based approximation rule in Section 3.3.

Remark (Synchronous vs. asynchronous control flow). In contrast to the Kantorovich distance operator $rpe(c,\ \mathcal{E})$, our compositional relational pre-expectation operator $\widetilde{rpe}(c, \mathcal{E})$ only gives useful (i.e., finite) bounds when the control flows in the two executions of c can be *synchronized*. For deterministic guards, this means that pairs of related executions always take the same branches; for randomized guards, this means that we can relate the random samplings so that pairs of related executions always take the same branches. In Section 8, we will extend our calculus to give more useful bounds when reasoning asynchronously.

Remark (Tightness of bounds). It is also complicated to estimate the exact loss between $\widetilde{rpe}(c, \mathcal{E})$ and $rpe(c, \mathcal{E})$, since lower bounds on $rpe(c, \mathcal{E})$ are not given by a witness coupling. Nonetheless, in our setting this limitation is not exclusive to our technique—in the statistical literature, lower bounds for stochastic processes are in general hard to compute and so the exact distance is often not known.

We now study the metatheory of our calculus. The main result is that our calculus is sound: it correctly upper bounds the Kantorovich distance.

Theorem 3.2 (Soundness of $\widetilde{rpe}$). *Let c be a* pWhile *program and $\mathcal{E} \in$ **Exp** be a relational expectation. Then $rpe(c, \mathcal{E}) \le \widetilde{rpe}(c, \mathcal{E})$, i.e., if $\widetilde{rpe}(c, \mathcal{E})(s_1, s_2) < \infty$ for $s_1, s_2 \in$ **State** then*

$$\mathbb{E}_{\mu_{s_1,s_2}}[\mathcal{E}] \le \widetilde{rpe}(c, \mathcal{E})(s_1, s_2) \quad \text{for some coupling} \quad \mu_{s_1,s_2} \in \Gamma([\![c]\!]s_1, [\![c]\!]s_2) .$$

Proof Sketch. By induction on c. Note that the theorem requires to show existence of a coupling that is below the $\widetilde{rpe}$. The most challenging cases are the ones for sampling and loops. The case for sampling follows from the following lemma, which is adapted from the theory of optimal transport [Villani 2008]:

Lemma 3.3. *Let $\mu_1, \mu_2 \in$ **Dist(State)** be two sub-distributions of countable support with the same weight, and let $\mathcal{E}$: **State**×**State** $\to \mathbb{R}^{\infty}_{\ge 0}$ be a relational expectation. There exists a coupling $\mu \in \Gamma(\mu_1, \mu_2)$ realizing the minimum Kantorovich distance:*

$$\mathbb{E}_{\mu}[\mathcal{E}] = \inf_{\mu \in \Gamma(\mu_1, \mu_2)} \mathbb{E}_{\mu}[\mathcal{E}] = \mathcal{E}^{\#}(\mu_1, \mu_2) .$$

The case for loops is challenging for another reason: it is not clear how to show that the pre-expectation operator is continuous in its second argument (but see Theorem 3.4). Instead, our proof relies on extracting a convergent sequence of couplings. Consider the following loop approximants:

$$c_0 \triangleq \textbf{while } tt \textbf{ do skip}$$
$$c_{i+1} \triangleq \textbf{if } e \textbf{ then } c; c_i \textbf{ else skip}$$

Each approximant executes at most i iterations of the loop; the zero-th approximant returns the zero distribution and does not execute any iterations of the loop body. We then define a sequence of pre-expectations corresponding to the approximants:

$$\mathcal{E}_0 \triangleq \widetilde{rpe}(c_0, \mathcal{E}) = 0$$
$$\mathcal{E}_{i+1} \triangleq \widetilde{rpe}(c_{i+1}, \mathcal{E}) = [e\langle 1\rangle \wedge e\langle 2\rangle] \cdot \widetilde{rpe}(c, \mathcal{E}_i) + [\neg e\langle 1\rangle \wedge \neg e\langle 2\rangle] \cdot \mathcal{E} + [e\langle 1\rangle \ne e\langle 2\rangle] \cdot \infty$$

It is not hard to see that also $\mathcal{E}_i = \Phi^i_{\mathcal{E},c}(0)$, but we cannot apply Kleene's fixpoint theorem to show that $\lim_{i\to\infty} \Phi^i_{\mathcal{E},c}(0) = \mathsf{lfp} X.\Phi_{\mathcal{E},c}(X)$, since we do not know if $\Phi_{\mathcal{E},c}$ is continuous. With this in mind, the bulk of the proof consists in showing: (i) For every i, $\Phi^i_{\mathcal{E},c}(0) \le \mathsf{lfp} X.\Phi_{\mathcal{E},c}(X)$ (ii) There exists a sequence of couplings $\mu_{i,s_1,s_2} \in \Gamma([\![c_i]\!]s_1, [\![c_i]\!]s_2)$ such that

$$\mathbb{E}_{\mu_{i,s_1,s_2}}[\mathcal{E}] \le \mathcal{E}_i(s_1, s_2) = \widetilde{rpe}(c_i, \mathcal{E}).$$

(iii) From the sequence μ_{i,s_1,s_2} we can extract a subsequence μ'_{i,s_1,s_2} that converges monotonically to a coupling $\tilde{\mu}_{s_1,s_2} \in \Gamma([\![\textbf{while } e \textbf{ do } c]\!]s_1, [\![\textbf{while } e \textbf{ do } c]\!]s_2)$ satisfying

$$\mathbb{E}_{(s_1',s_2')\sim\tilde{\mu}_{s_1,s_2}}[\mathcal{E}(s_1', s_2')] \le \widetilde{rpe}(\textbf{while } e \textbf{ do } c, \mathcal{E})(s_1, s_2).$$

$$\frac{\mathcal{E} \leq \mathcal{E}'}{\widetilde{rpe}(c,\mathcal{E}) \leq \widetilde{rpe}(c,\mathcal{E}')}\ \textsc{Mono} \qquad \frac{FV(\mathcal{E}') \cap MV(c) = \emptyset}{\widetilde{rpe}(c,\mathcal{E}+\mathcal{E}') \leq \widetilde{rpe}(c,\mathcal{E}) + \mathcal{E}'}\ \textsc{Const}$$

$$\frac{}{\widetilde{rpe}(c,\mathcal{E}) + \widetilde{rpe}(c,\mathcal{E}') \leq \widetilde{rpe}(c,\mathcal{E}+\mathcal{E}')}\ \textsc{SupAdd} \qquad \frac{f\colon \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0} \text{ linear, with } f(\infty) \triangleq \infty}{\widetilde{rpe}(c, f \circ \mathcal{E}) = f \circ \widetilde{rpe}(c,\mathcal{E})}\ \textsc{Scale}$$

$$\frac{M\colon \mathbf{State} \times \mathbf{State} \to \Gamma(\llbracket d \rrbracket, \llbracket d \rrbracket)}{\widetilde{rpe}(x \xleftarrow{\$} d, \mathcal{E}) \leq \mathbb{E}_{(v_1,v_2)\sim M(-,-)}[\mathcal{E}\{v_1, v_2/x\langle 1\rangle, x\langle 2\rangle\}]}\ \textsc{Samp}$$

$$\frac{f\colon \mathbf{State} \times \mathbf{State} \to (D \to D) \text{ bijection}}{\widetilde{rpe}(x \xleftarrow{\$} U(D), \mathcal{E}) \leq \frac{1}{|D|} \sum_{v \in D} \mathcal{E}\{v, f(-,-)(v)/x\langle 1\rangle, x\langle 2\rangle\}}\ \textsc{Unif}$$

$$\frac{[e\langle 1\rangle \wedge e\langle 2\rangle] \cdot \widetilde{rpe}(c,\mathcal{I}) + [\neg e\langle 1\rangle \wedge \neg e\langle 2\rangle] \cdot \mathcal{E} + [e\langle 1\rangle \neq e\langle 2\rangle] \cdot \infty \leq \mathcal{I}}{\widetilde{rpe}(\mathbf{while}\ e\ \mathbf{do}\ c, \mathcal{E}) \leq \mathcal{I}}\ \textsc{Inv}$$

Fig. 2. Properties of relational pre-expectation operator $\widetilde{rpe}(c,\mathcal{E})$.

□

It is natural to wonder whether our relational pre-expectation operator is continuous—this property is not needed for soundness, but it does hold for similar pre-expectation calculi. While we do not know whether continuity holds for *all* programs, it does hold for programs that sample from *finite* distributions. Note that such programs can still produce distributions with infinite support by sampling in a loop.

Theorem 3.4 (Continuity of $\widetilde{rpe}$). *Let c be a* pWhile *program where all primitive distributions have* finite *support, and let* $\mathcal{E}_n \in \mathbf{Exp}$ *for* $n \in \mathbb{N}$ *be a monotonically increasing chain of relational expectations converging pointwise to* $\mathcal{E} \in \mathbf{Exp}$. *Then,*

$$\widetilde{rpe}(c,\mathcal{E}) = \sup_{n \in \mathbb{N}} \widetilde{rpe}(c,\mathcal{E}_n).$$

Proof Sketch. By induction on the structure of c. The most challenging case is for sampling instructions, where the proof depends on the following continuity property for the Kantorovich distance that we establish for distributions with finite support:

Lemma 3.5. *Let* $\mu_1, \mu_2 \in \mathbf{Dist}(\mathbf{State})$ *be two distributions with finite support, and let* $\mathcal{E}_n\colon \mathbf{State} \times \mathbf{State} \to \mathbb{R}^{\infty}_{\geq 0}$ *be a monotonically increasing chain of relational expectations converging pointwise to* $\mathcal{E}\colon \mathbf{State} \times \mathbf{State} \to \mathbb{R}^{\infty}_{\geq 0}$. *Then:*

$$\inf_{\mu \in \Gamma(\mu_1,\mu_2)} \mathbb{E}_\mu[\mathcal{E}] = \inf_{\mu \in \Gamma(\mu_1,\mu_2)} \mathbb{E}_\mu[\lim_{n\to\infty} \mathcal{E}_n] = \lim_{n\to\infty} \inf_{\mu \in \Gamma(\mu_1,\mu_2)} \mathbb{E}_\mu[\mathcal{E}_n].$$

The proof of this lemma is involved, and extending it to more general distributions is out of scope. We defer the details to the full version. □

3.3 Reasoning with Relational Pre-expectations

The definition of $\widetilde{rpe}$ in Fig. 1 is sufficient to prove relational properties of probabilistic programs in theory, but there are some practical obstacles.

- Comparing different relational pre-expectations for the same program: using the definition to compute each relational pre-expectation separately is tedious.

- Computing the relational pre-expectation for random sampling: it requires computing a minimum over all couplings.
- Computing the relational pre-expectation for loops: it is often not possible to compute the least fixed point in closed form.

To make our operator easier to use, we introduce a collection of auxiliary properties in Fig. 2. We briefly describe the rules below.

Basic properties. The first four rules are basic properties of relational pre-expectations. Rule Mono states that the $\widetilde{rpe}$ transformer is monotone, and Const intuitively states that the relational pre-expectation of $\mathcal{E}$ is $\mathcal{E}$ if c doesn't modify $\mathcal{E}$; the rule is carefully stated to behave correctly when $\widetilde{rpe}(c, \mathcal{E})$ is infinite.

The next two rules encode linearity-like properties of relational pre-expectations. SupAdd states that the property is super-additive: the relational pre-expectation of a sum can be greater than the sum of the relational pre-expectations. Intuitively, this is because $\widetilde{rpe}(c, \mathcal{E})$ involves an infimum for random sampling, and the infimum of a sum at least as large as the sum of the infima. Scale states that the relational pre-expectation is preserved by scaling. The requirement that the scaling function satisfies $f(\infty) = \infty$ is needed for correctly handling scaling by 0: $\widetilde{rpe}(c, \mathcal{E})$ may be infinite, even if $\mathcal{E}$ is identically zero.

As expected, these rules are sound.

Theorem 3.6 (Soundness of basic rules). *Mono, Const, SupAdd, and Scale are sound.*

Proof. Proof sketch By induction on the program c; we defer details to the full verison. □

Bounding the pre-expectation for sampling. Using the Kantorovich distance for defining the relational pre-expectation of a sampling command $x \xleftarrow{\$} d$ is theoretically clean, but inconvenient in practice for two reasons. First, the set of couplings $\Gamma([\![x \xleftarrow{\$} d]\!]s_1, [\![x \xleftarrow{\$} d]\!]s_2)$ over which the infimum is computed is a set of distributions over pairs of states. Given denotations of primitive distributions $[\![d]\!] \in \mathbf{Dist}(D)$, it would be more convenient to reason about the set $\Gamma([\![d]\!], [\![d]\!])$—this is a set of distributions over pairs of sampled values $D \times D$, rather than pairs of memories. Second, computing the infimum is often difficult, and moreover unnecessary for establishing upper bounds.

Corresponding to the Samp rule, the following result states that we can actually upper bound this Kantorovich distance by picking *any* coupling of the primitive distribution with itself; we call such a function $M \colon \mathbf{State} \times \mathbf{State} \to \Gamma([\![d]\!], [\![d]\!])$ a *coupling function* (on d).

Proposition 3.7. *Let d be a primitive distribution, and let M be a coupling function on d. For any relational expectation $\mathcal{E} \in \mathbf{Exp}$, we have:*

$$\widetilde{rpe}(x \xleftarrow{\$} d, \mathcal{E}) \leq \mathbb{E}_{(v_1, v_2) \sim M(-,-)}[\mathcal{E}\{v_1, v_2 / x\langle 1\rangle, x\langle 2\rangle\}] .$$

We can reuse common couplings of primitive distributions across different proofs. For example, let D be a finite, non-empty set and let $f \colon \mathbf{State} \times \mathbf{State} \to (D \to D)$ map pairs of program states to bijections on D. Then the *bijection coupling M_f*, the coupling function on the uniform distribution $U(D)$ is defined by

$$f(s_1, s_2)(x_1, x_2) = \begin{cases} 1/|D| & : f(s_1, s_2)(x_1) = x_2 \\ 0 & : \text{otherwise} \end{cases} ,$$

where x_1 and x_2 are elements in D. Specialized to this case, Proposition 3.7 gives the rule Unif:

$$\begin{aligned}\widetilde{rpe}(x \xleftarrow{\$} U(D), \mathcal{E}) &\leq \widetilde{rpe}(x \xleftarrow{\$} d, \mathcal{E}) \leq \mathbb{E}_{(v_1,v_2)\sim M_f(-,-)}[\mathcal{E}\{v_1, v_2/x\langle 1\rangle, x\langle 2\rangle\}] \\ &\leq \mathbb{E}_{v\sim [\![U(D)]\!]}[\mathcal{E}\{v, f(-,-)(v)/x\langle 1\rangle, x\langle 2\rangle\}] \\ &= \frac{1}{|D|}\sum_{v\in D}\mathcal{E}\{v, f(-,-)(v)/x\langle 1\rangle, x\langle 2\rangle\} .\end{aligned}$$

Different coupling functions can give upper bounds of different strengths—selecting appropriate couplings to show the target property is the key part of reasoning by couplings. This technique is well-known to probability theory, where it is called the *coupling method* [Aldous 1983].

Bounding the pre-expectation for loops. As in the case of sampling, it may not always be desirable or possible to compute the fixed point for loops. Instead, we can upper bound the relational pre-expectation by a relational expectation $\mathcal{I}$, called an *invariant*—intuitively, if the relational pre-expectation of $\mathcal{I}$ with respect to the loop body is at most $\mathcal{I}$, then the relational pre-expectation of the loop is also at most $\mathcal{I}$. Formally, this reasoning is captured by Inv and the following theorem:

Theorem 3.8. *Let $\mathcal{I} \in$ **Exp** be a relational expectation. If*

$$[e\langle 1\rangle \wedge e\langle 2\rangle] \cdot \widetilde{rpe}(c, \mathcal{I}) + [\neg e\langle 1\rangle \wedge \neg e\langle 2\rangle] \cdot \mathcal{E} + [e\langle 1\rangle \neq e\langle 2\rangle] \cdot \infty \leq \mathcal{I},$$

then $\widetilde{rpe}(\textbf{while } e \textbf{ do } c, \mathcal{E}) \leq \mathcal{I}$.

Proof. Let Φ be the characteristic functional of the loop, as defined for the relational pre-expectation. The hypothesis implies $\Phi(\mathcal{I}) \leq \mathcal{I}$, so $\mathcal{I}$ is a prefixed point of Φ. By Park induction [Park 1969], the least fixed point $\widetilde{rpe}(\textbf{while } e \textbf{ do } c, \mathcal{E})$ is less than or equal to $\mathcal{I}$. □

3.4 Embedding $\mathbb{E}$pRHL

Expectation Probabilistic Relational Hoare Logic ($\mathbb{E}$pRHL) is a quantitative extension of pRHL [Barthe et al. 2018]. Judgments of $\mathbb{E}$pRHL are of the form: $\{P; \mathcal{E}\}\ c_1 \sim_f c_2\ \{Q; \mathcal{E}'\}$ where P, Q are boolean-valued assertions, $\mathcal{E}, \mathcal{E}'$ are relational expectations, f is an affine function of the form $ax + b$, where $a, b \in \mathbb{R}_{\geq 0}$, and c_1 and c_2 are pWhile programs. This judgment states that for every pair of input states s_1, s_2 satisfying the pre-condition P, there is a coupling μ of $[\![c_1]\!](s_1)$, $[\![c_2]\!](s_2)$ whose support lies within the post-condition Q, and moreover $\mathbb{E}_\mu[\mathcal{E}'] \leq f(\mathcal{E}(s_1, s_2))$. We can embed the core inference rules of $\mathbb{E}$pRHL in our proof system.

Theorem 3.9 (Embedding $\mathbb{E}$pRHL). *Let $\vdash \{P; \mathcal{E}\}\ c \sim_f c\ \{Q; \mathcal{E}'\}$ be a valid $\mathbb{E}$pRHL judgmentwith finite $\mathcal{E}$ and $\mathcal{E}'$. Then:*

$$\widetilde{rpe}(c, \mathcal{E}' + [\neg Q] \cdot \infty) \leq f(\mathcal{E}) + [\neg P] \cdot \infty.$$

Furthermore, this inequality can be derived using just the definition of $\widetilde{rpe}(c, \mathcal{E})$ for skip, assignment, sequence, and conditionals in Figure 1, *and the auxiliary proof rules in* Figure 2.

Intuitively, the bound on the relational pre-expectation captures the validity of the original $\mathbb{E}$pRHL judgment. For any pair of states (s_1, s_2), if (s_1, s_2) does not satisfy P, then the right-hand side is infinite and the bound trivially holds. If (s_1, s_2) satisfies P, then the right-hand side is finite (since $\mathcal{E}$ is finite) and the relational pre-expectation is finite. This implies that Q must be satisfied almost surely in the coupling and $\widetilde{rpe}(c, \mathcal{E}') \leq f(\mathcal{E})$. This last inequality recovers the $\mathbb{E}$pRHL judgment's bound on the output distance in terms of the input distance. Furthermore, the embedding shows that the bound is derivable in our calculus *without* computing infimums over couplings for sampling, or computing least fixed points for loops.

```
sgd(S)
    $w \leftarrow w_0$;
    $t \leftarrow 0$;
    while $t < T$ do
        $s \xleftarrow{\$} [S]$;
        $g \leftarrow \nabla \ell(s, -)(w)$;
        $w \leftarrow w - \alpha_t \cdot g$;
        $t \leftarrow t + 1$;
```

(a) Stochastic Gradient Descent (SGD)

```
TD0(V)
    $n \leftarrow 0$;
    while $n < N$ do
        $i \leftarrow 0$;
        while $i < |S|$ do
            $a \xleftarrow{\$} \pi(i); r \xleftarrow{\$} \mathcal{R}(i, a); j \xleftarrow{\$} \mathcal{P}(i, a)$;
            $W[i] \leftarrow (1 - \alpha) \cdot V[i] + \alpha \cdot (r + \gamma \cdot V[j])$;
            $i \leftarrow i + 1$
        $V \leftarrow W; n \leftarrow n + 1$;
```

(b) TD(0) learning algorithm

Fig. 3. Example programs: Stability and convergence

4 WARMUP EXAMPLE: STABILITY OF SGD

To demonstrate our relational pre-expectation operator, we analyze the stability of Stochastic Gradient Descent (SGD) as our warmup example. SGD is a core tool in modern machine learning; variants of SGD are commonly used in practice for training neural networks. Its stability was first established in Hardt et al. [2016], and it was later formalized in a relational program logic $\mathbb{E}$pRHL [Barthe et al. 2018]. While the $\mathbb{E}$pRHL proof involves complex proof rules, our calculus can establish the same property with significantly cleaner reasoning.

4.1 Background

Let Z be a space of labeled *examples*, e.g., images annotated with their main subject. A *learning algorithm* $A\colon S \to \mathbb{R}^d$ takes a set $S \in Z^n$ of examples as input and produces ("learns") *parameters* $w \in \mathbb{R}^d$ as output. The algorithm is tailored to a given *loss function* $\ell\colon Z \to \mathbb{R}^d \to [0, 1]$, which describes how well an example is labeled by some parameters. The goal is to find parameters that have low loss on examples.

In machine learning, *uniform stability* is a useful property for learning algorithms. In a nutshell, a randomized learning algorithm A is ϵ*-uniformly stable* if for all pairs S, S' of training sets differing in exactly one example, and for all examples $z \in Z$, the expected losses of z are close:

$$|\mathbb{E}_{A(S)}[\ell(z)] - \mathbb{E}_{A(S')}[\ell(z)]| \leq \epsilon \ .$$

Stable learning algorithms *generalize*: their performance on new, unseen examples is similar to their performance on the training set [Bousquet and Elisseeff 2002]. In particular, stability controls how much a learning algorithm can *overfit* the training set.

4.2 Verifying Stability for Stochastic Gradient Descent

We consider the program **sgd** in Figure 3a. The gradient ∇ is a higher-order function[1] with type $\nabla\colon (\mathbb{R}^d \to [0, 1]) \to (\mathbb{R}^d \to \mathbb{R}^d)$; we assume that it is well-defined and given. In SGD, the true gradient of a function is approximated by a gradient g at a single sample s. The step sizes α_t (with $t \in \mathbb{N}$) are a sequence of real numbers that control (together with the local gradient g) how to adjust the parameters in each iteration of SGD. Following Hardt et al. [2016], we make the following assumptions:

[1]This makes our states non-discrete, but the distributions over them will still have discrete support, since they are generated by a composition of discrete samplings.

(1) The loss function ℓ is convex and L-Lipschitz in its second argument, i.e., $|\ell(z,w) - \ell(z,w')| \leq L \cdot \|w - w'\|$ for all parameters $w, w' \in \mathbb{R}^d$.
(2) The gradient $\nabla\ell(z,-) \colon \mathbb{R}^d \to \mathbb{R}^d$ is β-Lipschitz for every $z \in Z$.
(3) The step sizes satisfy $0 \leq \alpha_t \leq 2/\beta$.

To show uniform stability, for any two training sets $S\langle 1\rangle, S\langle 2\rangle$ differing in one element and every example $z \in Z$, our proof obligation is

$$|\mathbb{E}_{\mathbf{sgd}(S\langle 1\rangle)}[\ell(z)] - \mathbb{E}_{\mathbf{sgd}(S\langle 2\rangle)}[\ell(z)]| \leq \gamma L \qquad \text{where } \gamma \triangleq \frac{2L}{n}\sum_{t=0}^{T-1}\alpha_t\,.$$

Rather than working with the loss function directly, we will first bound the pre-expectation of the distance $\|w\langle 1\rangle - w\langle 2\rangle\|$ and then use the L-Lipschitz property of ℓ to conclude uniform stability. As usual, the main part of the proof is bounding the pre-expectation of the loop. We use the following loop invariant:

$$\mathcal{I} \triangleq [t\langle 1\rangle \neq t\langle 2\rangle] \cdot \infty + [t\langle 1\rangle = t\langle 2\rangle] \cdot \left(\|w\langle 1\rangle - w\langle 2\rangle\| + \frac{2L}{n}\sum_{j=t\langle 1\rangle}^{T-1}\alpha_j \right).$$

By the loop rule (Theorem 3.8), it suffices to show the following invariant condition:

$$[e\langle 1\rangle \wedge e\langle 2\rangle] \cdot \widetilde{rpe}(bd, \mathcal{I}) + [\neg e\langle 1\rangle \wedge \neg e\langle 2\rangle] \cdot \|w\langle 1\rangle - w\langle 2\rangle\| + [e\langle 1\rangle \neq e\langle 2\rangle] \cdot \infty \quad \leq \quad \mathcal{I}\,. \tag{1}$$

The main case corresponds to the first term, where both loop guards $e\langle 1\rangle$ and $e\langle 2\rangle$ are true. To bound the pre-expectation $\widetilde{rpe}(bd, \mathcal{I})$, we consider $\widetilde{rpe}(bd, \mathcal{I}) = \widetilde{rpe}(s \xleftarrow{\$} U(S), \mathcal{I}')$ where

$$\mathcal{I}' \triangleq [t\langle 1\rangle{+}1 \neq t\langle 2\rangle{+}1] \cdot \infty + [t\langle 1\rangle{+}1 = t\langle 2\rangle{+}1] \cdot P, \text{ with}$$
$$P \triangleq \frac{2L}{n}\sum_{j=t\langle 1\rangle+1}^{T-1}\alpha_j + \left\| \begin{matrix} (w\langle 1\rangle - \alpha_{t\langle 1\rangle}\nabla\ell(s\langle 1\rangle,-)(w\langle 1\rangle)) \\ -(w\langle 2\rangle - \alpha_{t\langle 2\rangle}\nabla\ell(s\langle 2\rangle,-)(w\langle 2\rangle)) \end{matrix} \right\|.$$

To handle the random sampling command, we apply the sampling rule (Proposition 3.7) with the coupling function M for the two uniform distributions $[S\langle 1\rangle]$ and $[S\langle 2\rangle]$ induced by the bijection $f \colon S\langle 1\rangle \to S\langle 2\rangle$ mapping the differing example in $S\langle 1\rangle$ to its counterpart in $S\langle 2\rangle$, and fixing all other examples. We then have $\widetilde{rpe}(s \xleftarrow{\$} U(S), \mathcal{I}') \leq \mathcal{I}''$, where

$$\mathcal{I}'' \triangleq [t\langle 1\rangle{+}1 \neq t\langle 2\rangle{+}1] \cdot \infty + [t\langle 1\rangle{+}1 = t\langle 2\rangle{+}1] \cdot P', \text{ with}$$
$$P' = \frac{2L}{n}\sum_{j=t\langle 1\rangle+1}^{T-1}\alpha_j + \frac{1}{n}\sum_{s\in S\langle 1\rangle}^{n-1} \left\| \begin{matrix} (w\langle 1\rangle - \alpha_{t\langle 1\rangle}\nabla\ell(s,-)(w\langle 1\rangle)) \\ -(w\langle 2\rangle - \alpha_{t\langle 2\rangle}\nabla\ell(f(s),-)(w\langle 2\rangle)) \end{matrix} \right\|$$

We focus on the terms of the last sum. Using the L-Lipschitz property of ℓ, when s is the differing example, we can bound the absolute difference by $\|w\langle 1\rangle - w\langle 2\rangle\| + 2\alpha_{t\langle 1\rangle}L$. When s is not the differing example, we have $s\langle 1\rangle = s\langle 2\rangle$. By the β-Lipschitz property of $\nabla\ell$, convexity, and $0 \leq \alpha_t \leq 2/\beta$, we can bound each of the terms by $\|w\langle 1\rangle - w\langle 2\rangle\|$. Combining the two cases gives

$$\widetilde{rpe}(bd, \mathcal{I}) \leq \left(\|w\langle 1\rangle - w\langle 2\rangle\| + \frac{2L}{n}\sum_{j=t\langle 1\rangle}^{T-1}\alpha_j \right)$$

for all input states with $t\langle 1\rangle = t\langle 2\rangle$ and $e\langle 1\rangle \wedge e\langle 2\rangle$. This establishes (1). Theorem 3.8 gives

$$\widetilde{rpe}(\textbf{while } e \textbf{ do } bd, \|w\langle 1\rangle - w\langle 2\rangle\|) \leq \mathcal{I}\,.$$

Finally, taking the pre-expectations of both sides with respect to the initial assignments yields

$$\widetilde{rpe}(\mathbf{sgd}(S), \|w\langle 1\rangle - w\langle 2\rangle\|) \leq \frac{2L}{n} \sum_{j=0}^{T-1} \alpha_j = \gamma,$$

when $S\langle 1\rangle$ and $S\langle 2\rangle$ differ in exactly one training example. Since ℓ is L-Lipschitz, we conclude

$$\widetilde{rpe}(\mathbf{sgd}(S), |\ell(z,w)\langle 1\rangle - \ell(z,w)\langle 2\rangle|) \leq \gamma L\,,$$

for any example $z \in Z$. By Theorem 2.4, the expected losses are at most γL apart:

$$|\mathbb{E}_{\mathbf{sgd}(S\langle 1\rangle)}[\ell(z)] - \mathbb{E}_{\mathbf{sgd}(S\langle 2\rangle)}[\ell(z)]| \leq \gamma L\,,$$

and so SGD satisfies γL-uniform stability.

Remark. This stability bound for SGD was previously verified in the program logic 𝔼pRHL [Barthe et al. 2018], using a complex rule for sequential composition (SeqCase) that required bounding the probability of selecting two differing examples. Our proof using $\widetilde{rpe}$ is much simpler, involving just compositional reasoning for sequencing and a loop invariant.

Remark. While our calculus was designed for probabilistic programs, it is also a useful tool for proving relational properties of deterministic programs. In the full version, we show how to prove a sensitivity bound for *projected gradient descent*, a deterministic version of SGD.

5 EXAMPLE: CONVERGENCE OF REINFORCEMENT LEARNING ALGORITHMS

In the previous section, the stability guarantee weakens as the program progresses: starting from two initially-equal parameter settings, the learned parameters may drift apart as SGD runs for more iterations. In the following two sections, we will apply our technique to prove a different style of guarantee: *probabilistic convergence* of two outputs, starting from two different inputs. Our first example shows convergence for a classical algorithm from Reinforcement Learning (RL) [Sutton 1988], guided by a novel analysis by Amortila et al. [2020].

5.1 Background

In the standard reinforcement learning setting, an agent (i.e., the learning algorithm) repeatedly interacts with the environment, a Markov Decision Process (MDP) with *state* space $\mathcal{S}$. At each step, the agent chooses an *action* from a set $\mathcal{A}$. The MDP draws a numeric *reward* according to a function $\mathcal{R} \colon \mathcal{S} \times \mathcal{A} \to \mathbf{Dist}([0, R])$, and transitions to a new random state drawn from a *transition* function $\mathcal{P} \colon \mathcal{S} \times \mathcal{A} \to \mathbf{Dist}(\mathcal{S})$. The current state of the process is known to the learner—imagine the current position of a chessboard—but the exact reward and transition functions $(\mathcal{R}, \mathcal{P})$ are not known. Given black-box access to $\mathcal{R}$ and $\mathcal{S}$, the goal of the learner is to find a *policy map* $\pi \colon \mathcal{S} \to \mathcal{A}$ that maximizes the learner's expected reward when interacting with the unknown MDP over an infinite time horizon; estimated rewards in the future are reduced by a discount factor $\gamma \in [0, 1)$ for each step into the future.

For many approaches to learning the optimal policy, an important requirement is estimating the *value function* $V \colon \mathcal{S} \to [0, R]$ of the MDP, i.e., the expected reward at each state if the agent were to repeatedly act according to some given policy π. *Temporal difference (TD)* learning is one approach to estimating the value function [Sutton 1988]. In brief, a TD learner maintains an estimate of V and loops through states in $\mathcal{S}$. At each state s, the learner selects an action $a \sim \pi(s)$, draws a reward $r \sim \mathcal{R}(s, a)$, and draws a transition $s' \sim \mathcal{P}(s, a)$. Then, the estimate $V(s)$ is updated by incorporating the observed reward r and the estimated value $V(s')$ of the new state.

Figure 3b shows one basic example of TD learning, known as TD(0). We assume that the program takes only one argument V, the initial estimate of the value function. All other parameters are assumed to be fixed: the current policy π, the reward and transition functions $\mathcal{R}$ and $\mathcal{P}$, the discount factor γ, the step size $\alpha \in (0,1)$—higher α allows V to evolve faster–and the number of iterations N.

5.2 Verifying Convergence for TD0

Since the true value function is not known, the initial estimate V is chosen with little information. A natural question is: does the algorithm converge to the same distribution no matter how V is initialized? If so, how fast does convergence happen, as a function of the number of iterations N? To answer these questions, we will verify that **TD0** is contractive on V. More specifically, we will show the quantitative bound

$$\widetilde{rpe}(\mathbf{TD0}(V), \|V\langle 1\rangle - V\langle 2\rangle\|_\infty) \le k^N \cdot \|V\langle 1\rangle - V\langle 2\rangle\|_\infty, \tag{2}$$

where $k \triangleq (1-\alpha+\alpha\gamma) < 1$. Before we describe the verification, we unpack the guarantee. First, the ∞-norms are defined by $\|V\langle 1\rangle - V\langle 2\rangle\|_\infty \triangleq \max_{i<|\mathcal{S}|} |V\langle 1\rangle[i] - V\langle 2\rangle[i]|$. By Theorem 3.2, Eq. (2) implies that for any inputs V_1 and V_2, there exists a coupling μ of the output distributions μ_1 and μ_2 from $\mathbf{TD0}(V\langle 1\rangle)$ and $\mathbf{TD0}(V\langle 2\rangle)$, such that:

$$\begin{aligned}
k^N \cdot \|V_1 - V_2\|_\infty &\ge \mathbb{E}_{(s_1,s_2)\sim\mu}[\|s_1(V) - s_2(V)\|_\infty] \\
&\ge \max_{i<|\mathcal{S}|} \mathbb{E}_{(s_1,s_2)\sim\mu}[|\, s_1(V[i]) - s_2(V[i])\,|] \\
&\ge \max_{i<|\mathcal{S}|} \left|\mathbb{E}_{(s_1,s_2)\sim\mu}[s_1(V[i]) - s_2(V[i])]\right| \\
&= \max_{i<|\mathcal{S}|} \left|\mathbb{E}_{s_1\sim\mu_1}[s_1(V[i])] - \mathbb{E}_{s_2\sim\mu_2}[s_2(V[i])]\right| \qquad \text{(by Theorem 2.4)}
\end{aligned}$$

In words, the right-hand side of the final line is the maximum difference between the average estimates of $V[i]$ in the two outputs, taking the maximum over all indices i. Since $k < 1$, both sides tend to zero exponentially quickly from any pair of starting states V_1 and V_2.

Inner loop. We start by analyzing the inner loop w_{in}. We first show that

$$\widetilde{rpe}(w_{in}, \|W\langle 1\rangle - W\langle 2\rangle\|_\infty) \le \mathcal{I}_{in}$$

for the invariant $\mathcal{I}_{in}$:

$$\begin{aligned}
\mathcal{I}_{in} \triangleq\ & [i\langle 1\rangle \ne i\langle 2\rangle] \cdot \infty \\
& + [i\langle 1\rangle = i\langle 2\rangle] \cdot \max_{l<|S|}([l < i\langle 1\rangle] \cdot |W\langle 1\rangle[l] - W\langle 2\rangle[l]| + [i\langle 1\rangle \le l] \cdot k \cdot \|V\langle 1\rangle - V\langle 2\rangle\|_\infty).
\end{aligned}$$

Let c_{in} be the body, and c_{samp} be the three sampling statements. Applying Inv, it suffices to show:

$$[i\langle 1\rangle < |\mathcal{S}| \wedge i\langle 2\rangle < |\mathcal{S}|]\cdot\widetilde{rpe}(c_{in}, \mathcal{I}_{in}) + [i\langle 1\rangle \ge |\mathcal{S}| \wedge i\langle 2\rangle \ge |\mathcal{S}|]\cdot\|W\langle 1\rangle - W\langle 2\rangle\|_\infty + [i\langle 1\rangle \ne i\langle 2\rangle]\cdot\infty \le \mathcal{I}_{in}$$

The main case is bounding $\widetilde{rpe}(c_{in}, \mathcal{I}_{in})$; the other cases are simpler. We describe the overall idea here, deferring details to the full version. To bound the relational pre-expectation for the three sampling instructions, we apply the sampling rule Samp. Since the relational pre-expectation is computed in reverse order, we must choose a coupling for sampling j first, then choose a coupling for sampling r, and then finally choose a coupling for sampling a. We aim to take the identity coupling in each case, ensuring $j\langle 1\rangle = j\langle 2\rangle$, $r\langle 1\rangle = r\langle 2\rangle$, and $a\langle 1\rangle = a\langle 2\rangle$, but there is a small problem: we can only take the identity coupling when samples are taken from the same distributions, i.e., $\mathcal{R}(i\langle 1\rangle, a\langle 1\rangle) = \mathcal{R}(i\langle 2\rangle, a\langle 2\rangle)$. The invariant assumes $i\langle 1\rangle = i\langle 2\rangle$, but we can only ensure $a\langle 1\rangle = a\langle 2\rangle$ after we have specified the couplings for j and r. Accordingly, our coupling functions for Samp will be of the following form: if $a\langle 1\rangle = a\langle 2\rangle$ then we take the identity coupling, otherwise we take the trivial (independent) coupling.

Outer loop. We now turn to the analysis of the outer loop. Consider the invariant:

$$\mathcal{I}_{out} \triangleq [n\langle 1\rangle \neq n\langle 2\rangle] \cdot \infty + [n\langle 1\rangle = n\langle 2\rangle] \cdot k^{(N \ominus n\langle 1\rangle)} \|V\langle 1\rangle - V\langle 2\rangle\|_\infty ,$$

where $N \ominus n$ denotes $\max(N-n, 0)$. We compute:

$$\begin{aligned}
&\widetilde{rpe}(i \leftarrow 0; w_{in}; V \leftarrow W; n \leftarrow n+1, \mathcal{I}_{out}) \\
&= \widetilde{rpe}(i \leftarrow 0; w_{in}, [n\langle 1\rangle \neq n\langle 2\rangle] \cdot \infty + [n\langle 1\rangle = n\langle 2\rangle] \cdot k^{(N \ominus (n\langle 1\rangle + 1))} \|W\langle 1\rangle - W\langle 2\rangle\|_\infty) \\
&\leq \widetilde{rpe}(i \leftarrow 0, [n\langle 1\rangle \neq n\langle 2\rangle] \cdot \infty + [n\langle 1\rangle = n\langle 2\rangle] \cdot k^{(N \ominus (n\langle 1\rangle + 1))} \cdot \mathcal{I}_{in}) \\
&\leq [n\langle 1\rangle \neq n\langle 2\rangle] \cdot \infty + [n\langle 1\rangle = n\langle 2\rangle] \cdot k \cdot k^{(N \ominus (n\langle 1\rangle + 1))} \|V\langle 1\rangle - V\langle 2\rangle\|_\infty = \mathcal{I}_{out}
\end{aligned}$$

where the last step holds because $\mathcal{I}_{in} = k \cdot \|V\langle 1\rangle - V\langle 2\rangle\|_\infty$ when $i = 0$. This establishes the outer invariant. Computing the pre-expectation of the first initialization, we conclude:

$$\widetilde{rpe}(\mathbf{TD0}(V), \|V\langle 1\rangle - V\langle 2\rangle\|_\infty) \leq k^N \cdot \|V\langle 1\rangle - V\langle 2\rangle\|_\infty .$$

6 EXAMPLE: RANDOM WALKS AND CARD SHUFFLES

In this section, we verify more challenging examples of probabilistic convergence from the theory of Markov chains, formalizing arguments by Aldous [1983] in his seminal work introducing the coupling method. Our use of relational pre-expectations is similar in spirit to the previous section, but there are two key differences: (1) we aim to prove convergence under Total Variation (TV) distance, which is the standard notion of distance in this field, and (2) our arguments will require selecting more complex couplings, instead of just the identity coupling.

6.1 Preliminaries: Card Shuffling and Markov Chain Mixing

For instance, consider the uniform distribution over all permutations of a deck of playing cards. To sample from this distribution—i.e., perform a *perfect shuffle*—we can implement a card shuffle algorithm that executes a sequence of simple randomized steps (e.g., swapping pairs of cards) and hope that after some number of steps, we will produce a shuffle that is close to uniform.

Abstracting a bit, card shuffling algorithms are a representative example of random walks for approximating complex distributions. This is a technique with a long history, combining elements of probability theory with statistical physics; and it is the basis of many heuristic algorithms used today, e.g., Markov Chain Monte Carlo (MCMC). From a theoretical perspective, the central question is: *how fast do these processes converge to their target distribution*? How many steps do we need to get within ϵ distance of the uniform distribution on shuffles?

Random walks and card shuffling algorithms are classical examples of *Markov chains*. A finite, discrete-time Markov chain is defined by a finite state space Σ and a transition function $P \colon \Sigma \to \mathbf{Dist}(\Sigma)$. Given an initial state σ, the associated Markov process $\{X_k^\sigma\}_{k \in \mathbb{N}}$ is a sequence of distributions such that $X_0^\sigma = \delta(\sigma)$ and $X_{k+1}^\sigma(\tau') = \sum_\tau X_k^\sigma(\tau) \cdot P(\tau, \tau')$. For example, the state space Σ could be the set of all permutations of a deck of cards, and the transition function τ could describe randomly splitting the deck and interleaving the halves.

Consider the TV distance $v(k)$ between two state distributions after running k steps from two states σ, τ, i.e., $v(k) \triangleq \max_{\sigma,\tau} TV(X_k^\sigma, X_k^\tau)$. If $v(k)$ tends to 0, then there exists a unique *stationary* distribution η such that $\eta(\sigma) \cdot P(\sigma, \sigma') = \eta(\sigma')$; typically, η will be the target distribution we are trying to sample from. Furthermore, $v(k)$ provides an upper bound on the distance between the state distribution after k steps to the stationary distribution η:

$$\max_\sigma TV(X_t^\sigma, \eta) \leq v(k) .$$

While it is usually not possible to derive $v(k)$ exactly, we can upper-bound $v(k)$ by constructing couplings of (X_t^σ, X_t^τ) and applying Theorems 2.3 and 2.5. In this way, we can prove bounds on the number of steps needed to get within some distance of the target distribution.

6.2 Warmup: Hypercube Walk

We start off with a simple random process for sampling N uniformly random bits, which serves as a toy version of the more complex random walks we will see later. Our *position* is a string of N bits (which can be regarded as a vertex of an N-dimensional *hypercube*). On every iteration of the walk we uniformly sample from $\{0, \ldots, N\}$. Note that there are $N+1$ possible draws, but only N coordinates: if we sample 0, then we do not move, otherwise we reverse the sampled coordinate i in the current position. We will show that starting from any two positions, the process *mixes rapidly*, i.e. starting from any position we will quickly reach the uniform distribution over positions.

Let $e(i) = (0, \ldots, 1, \ldots, 0) \in \{0,1\}^N$ be the position where all coordinates are set to zero except for coordinate i, which is set to one. We also write $\oplus$ for xor applied coordinate-wise. We can model K steps of the random walk with the following simple pWhile program:

```
**hWalk**($pos, N, K$)
  $k \leftarrow 0$;
  **while** $k < K$ **do**
    $i \xleftarrow{\$} U([N{+}1])$;
    **if** $i \neq 0$ **then** $pos \leftarrow pos \oplus e(i)$;
    $k \leftarrow k + 1$
```

Consider two program runs, started at $pos\langle 1\rangle$ and $pos\langle 2\rangle$ respectively. Let d_H be normalized Hamming distance between the two positions:

$$d_H \triangleq \frac{1}{N}\sum_{i=1}^{N}[pos\langle 1\rangle[i] \neq pos\langle 2\rangle[i]] \,.$$

That is, d_H equals the fraction of coordinates where $pos\langle 1\rangle$ and $pos\langle 2\rangle$ differ. Let $C(pos\langle 1\rangle, pos\langle 2\rangle) \subseteq [N]$ be the set of differing coordinates. We specify a coupling on $U([N{+}1])$ by giving a bijection on $[N{+}1]$. There are three cases:

(1) $d_H \geq 2/N$: Let $C(pos\langle 1\rangle, pos\langle 2\rangle) = \{i_0, \ldots, i_{m-1}\}$. Take the bijection that behaves like the identity on $[N{+}1] \setminus C(pos\langle 1\rangle, pos\langle 2\rangle)$ and that, for all $0 \leq n \leq m$, maps i_n to i_{n+1}, where we set $i_m = i_0$.
(2) $d_H = 1/N$: Take the bijection exchanging the differing coordinate and 0.
(3) $d_H = 0$: Take the identity bijection.

The coupling captures the following intuition. When $d_H \geq 2/N$, the distance decreases by $2/N$ if we select a differing coordinate; otherwise, it remains unchanged. Likewise when $d_H = 1/N$, if we select the differing coordinate or 0, then the distance decreases by $1/N$ (to 0); otherwise, the distance remains unchanged.

We can analyze the program **hWalk** using our relational pre-expectation calculus. Let the target relational expectation be d_H. The main step in the reasoning is to select a relational invariant for the loop. We define:

$$\mathcal{I} \triangleq [k\langle 1\rangle \neq k\langle 2\rangle] \cdot \infty + [k\langle 1\rangle = k\langle 2\rangle] \cdot d_H \cdot \left(\frac{N-1}{N+1}\right)^{K \ominus k\langle 1\rangle} .$$

```
rTop(deck, N, K)
  k ← 0;
  while k < K do
    p ←$ U([N]);
    deck ← shiftR(deck, p);
    k ← k + 1;
```

```
rTrans(deck, N, K)
  k ← 0;
  while k < K do
    p ←$ U([N]); p′ ←$ U([N]);
    c ← deck[p]; c′ ← deck[p′];
    deck[p] ← c′; deck[p′] ← c;
    k ← k + 1;
```

```
riffle(deck, N, K)
  k ← 0;
  while k < K do
    b ←$ U({0, 1}^N);
    top ← deck(b̄);
    bot ← deck(b);
    deck ← cat(top, bot);
    k ← k + 1;
```

Fig. 4. Shuffling algorithms

Then, we can verify for the loop **while** $k < K$ **do** bd of program **hWalk** that

$$\begin{array}{rl} & [(k\langle 1\rangle < K\langle 1\rangle) \wedge (k\langle 2\rangle < K\langle 2\rangle)] \cdot \widetilde{rpe}(bd, \mathcal{I}) \\ + & [(k\langle 1\rangle \geq K\langle 1\rangle) \wedge (k\langle 2\rangle \geq K\langle 2\rangle)] \cdot d_H \\ + & [(k\langle 1\rangle < K\langle 1\rangle) \neq (k\langle 2\rangle < K\langle 2\rangle)] \cdot \infty \qquad \leq \quad \mathcal{I}, \end{array}$$

and conclude by the loop rule (Theorem 3.8):

$$\widetilde{rpe}(\textbf{while } k < K \textbf{ do } bd, d_H) \leq \mathcal{I}.$$

The main step here is showing that

$$[(k\langle 1\rangle < K\langle 1\rangle) \wedge (k\langle 2\rangle < K\langle 2\rangle)] \cdot \widetilde{rpe}(bd, \mathcal{I}) \;\leq\; [(k\langle 1\rangle < K\langle 1\rangle) \wedge (k\langle 2\rangle < K\langle 2\rangle)] \cdot \mathcal{I}\ ,$$

where we use the fact that the coupling described above makes d_H decrease.

Pushing the invariant past the initialization instruction $k \leftarrow 0$ yields:

$$\widetilde{rpe}(\textbf{hWalk}(pos, N, K), d_H) \leq \widetilde{rpe}(k \leftarrow 0, \mathcal{I}) = \left(\frac{N-1}{N+1}\right)^K .$$

Since the distance d_H takes distance at least $1/N$ on pairs of distinct positions, by Theorem 2.5 the TV distance between the distributions over positions satisfies

$$v(K, N) = \max_{p_1, p_2 \in \{0,1\}^N} TV([\![\textbf{hWalk}]\!](p_1, N, K), [\![\textbf{hWalk}]\!](p_2, N, K)) \leq N\left(1 - \frac{2}{N+1}\right)^K .$$

Plugging in specific values gives concrete bounds between the two output distributions. Let $\rho > 1$. To achieve a bound of $O(1/\rho)$ on the right hand side, we need to take $K \geq (1/2)N\log(N\rho)$. The inequality above also gives useful asymptotic information; if we set $\rho = N$, and take $K \geq N \log N$, the right-hand side is asymptotically bounded by $O(1/N)$ for large N. We can show that this converges to the uniform distribution over vectors. We provide more details in Section 7. In summary, we have shown the following:

Theorem 6.1. *Let $K = N \log N$. For any initial position pos,*

$$TV\left(\textbf{hWalk}(pos, N, K), U(\{0,1\})^N)\right) \in O(1/N)\ .$$

6.3 Random-to-Top Shuffle

For our shuffling examples, we will need some notation. We view a permutation *deck* as a map from positions in $p \in [N]$ to names of cards in $c \in C$, and often take C to be $[N]$. We let $deck[p]$ denote the card at position p, while $deck^{-1}(c)$ denotes the position corresponding to card c. Summation over an empty set of indices is treated as zero, while the product over an empty set of indices is treated as one. We outline the arguments here.

For our first card shuffling algorithm we consider the *random-to-top* shuffle. In each iteration, it selects a random position in the deck and moves the card at that position to the top.[2] We model this shuffle with program **rTop** in Figure 4. For a given input deck of size N, the program repeats K times the process of selecting a random card and moving it to the top. The operation $\mathsf{shiftR}(deck, j)$ takes the block $deck[0], \ldots, deck[j]$ and cycles it one position to the right (thus moving $deck[j]$ to the top), leaving the rest of the deck intact.

We are interested in bounding the distance between the stationary distribution—which in this case is the uniform distribution—and the output distribution after K iterations. We will start with two decks of size N that are both permutations of $[N]$. As in the hypercube example, we bound the relational pre-expectation of the normalized Hamming distance:

$$d_H \triangleq \frac{1}{N} \sum_{i=0}^{N-1} [deck\langle 1\rangle[i] \neq deck\langle 2\rangle[i]] \,.$$

Note that d_H takes distance at least $1/N$ on pairs of distinct permutations. If we can show that the relational pre-expectation of d_H is not too big, then we can apply Theorem 2.5 to conclude that the final distributions over permutations have a close TV distance. It will be convenient to work with an auxiliary distance:

$$d_M \triangleq (1/N) \cdot \left(N - \max_i \left(\forall j < i. deck\langle 1\rangle[j] = deck\langle 2\rangle[j]\right)\right) \,.$$

The idea is that the coupling chooses identical cards on both decks and moves them to the top. This will form a block of matched cards on the top of both decks. Intuitively, d_M measures the fraction of the deck that is not part of this top block. The target distance d_H is upper-bounded by d_M, since d_M counts all cards outside the first block as different. Bounds on d_H follow from bounds on d_M. To bound the relational pre-expectation of d_M, we take the invariant:

$$\mathcal{I} \triangleq [k\langle 1\rangle \neq k\langle 2\rangle] \cdot \infty + [k\langle 1\rangle = k\langle 2\rangle] \cdot d_M \cdot \left(\frac{N-1}{N}\right)^{K \ominus k\langle 1\rangle} \,.$$

We can check that it satisfies the inequality

$$[k\langle 1\rangle < K \wedge k\langle 2\rangle < K] \cdot \widetilde{rpe}(bd, \mathcal{I}) + [k\langle 1\rangle \geq K \wedge k\langle 2\rangle \geq K] \cdot d_H + [(k\langle 1\rangle < K) \neq (k\langle 2\rangle < K)] \cdot \infty \leq \mathcal{I},$$

where bd is the loop body. The main case is to show the inequality for the first term when both loop guards are true: we need to bound the relational pre-expectation of $\mathcal{I}$ with respect to bd. We can bound

$$\widetilde{rpe}(bd, \mathcal{I}) \leq d_M \cdot \left(\frac{N-1}{N}\right)^{K \ominus k\langle 1\rangle} ,$$

by applying the sampling rule (Proposition 3.7) with the coupling function M that selects the same card in both decks:

$$M(s_1, s_2)(p_1, p_2) \triangleq \begin{cases} 1/N & : [\![deck]\!] s_1[p_1] = [\![deck]\!] s_2[p_2] \\ 0 & : \text{otherwise.} \end{cases}$$

The idea is that if we pick two cards in the first matched block, which happens with probability $(1 - d_M)$, then the distance will remain the same. Otherwise, we will create at least one new matched pair in the first block and the distance will decrease by $1/N$. Hence, we can apply the loop rule (Theorem 3.8) to conclude:

$$\widetilde{rpe}(\textbf{while } k < K \textbf{ do } bd, d_H) \leq \mathcal{I} \,.$$

[2]This algorithm is the time-reversed version of the *top-to-random* shuffle, where the top card is moved to a random position. It is known that a Markov chain's convergence behavior is equivalent to that of its reversed process [Aldous 1983].

Computing the relational pre-expectation of $\mathcal{I}$ with respect to the first instruction, we have

$$\widetilde{rpe}(\mathbf{rTop}(deck, N, K), d_H) \leq \left(\frac{N-1}{N}\right)^K ,$$

noting that the distance d_M between the initial decks is at most 1. Since d_H assigns pairs of distinct decks a distance at least $1/N$, Theorem 2.5 implies that the TV distance between the distributions over decks satisfies:

$$v(K, N) = \max_{d_1, d_2 \in [N]} TV([\![\mathbf{rTop}]\!](d_1, N, K), [\![\mathbf{rTop}]\!](d_2, N, K)) \ \leq N \left(\frac{N-1}{N}\right)^K .$$

For example, if we choose K to be $N \log(N\rho)$, then the distance between permutation distributions is bounded by $O(1/\rho)$ for large N and $\rho > 1$. By setting $\rho = N$, we have shown the following:

Theorem 6.2. *Let $K = 2N \log N$, and $Perm([N])$ be the set of permutations over N. For any initial permutation of deck,*

$$TV(\mathbf{rTop}\,(deck, N, K), U(Perm([N]))) \in O(1/N).$$

6.4 Random Transpositions Shuffle

Our next shuffle (**rTrans** in Figure 4) repeatedly selects two positions uniformly at random and swaps the cards, allowing for the possibility of swapping a card with itself. As before, let d_H be the normalized Hamming distance between the two decks. We aim to bound $\widetilde{rpe}(\mathbf{rTrans}, d_H)$. As before, the key of the proof is finding an invariant for the loop. We take:

$$\mathcal{I} \triangleq [k\langle 1\rangle \neq k\langle 2\rangle] \cdot \infty + [k\langle 1\rangle = k\langle 2\rangle] \cdot d_H \cdot \left(1 - \frac{1}{N^2}\right)^{K \ominus k\langle 1\rangle}$$

There are two samplings in the loop body, so we need to provide two couplings. For the first sampling p, we use the identity coupling. For the second sampling p', we couple using the bijection induced by the two decks $deck\langle 1\rangle$ and $deck\langle 2\rangle$, i.e., the coupling matches every position $p'\langle 1\rangle$ with the unique position $p'\langle 2\rangle$ such that $deck[p']\langle 1\rangle = deck[p']\langle 2\rangle$. There are three cases: (1) if cards at $p\langle 1\rangle, p\langle 2\rangle$ are already matched, d_H remains unchanged; (2) if positions $p'\langle 1\rangle, p'\langle 2\rangle$ are equal, d_H remains unchanged; otherwise (3) d_H decreases by 1. This is enough to show that the invariant decreases. We can conclude:

$$\widetilde{rpe}(\mathbf{rTrans}(deck, N, K), d_H) \leq \left(1 - \tfrac{1}{N^2}\right)^K$$

using the fact that d_H between the inputs is at most 1. Since d_H takes value of at least $1/N$ for pairs of distinct decks, by Theorem 2.5

$$v(K, N) = \max_{d_1, d_2 \in [N]} TV([\![\mathbf{rTrans}]\!](d_1,N,K), [\![\mathbf{rTrans}]\!](d_2,N,K)) \ \leq \ N\left(1 - \frac{1}{N^2}\right)^K ,$$

so the distance between the deck distribution and the uniform distribution decreases as K increases. If we take $K \geq N^2 \log(N\rho)$, then the right-hand side is bounded asymptotically by $O(1/\rho)$ for large N. By setting $\rho = N$, we conclude:

Theorem 6.3. *Let $K = 2N^2 \log N$, and $Perm([N])$ be the set of permutations over N. For any initial permutation of deck,*

$$TV(\mathbf{rTrans}(deck, N, K), U(Perm([N]))) \in O(1/N).$$

Remark. Aldous' [Aldous 1983] bound is slightly sharper: the TV distance between output distributions is bounded by $O(1/N)$ asymptotically already for $K \geq CN^2$ for some constant C. This discrepancy appears because our proofs are carried out compositionally, while Aldous uses a global analysis. However, it is possible that a clever choice of coupling or loop invariant could let us match Aldous' bound.

6.5 Uniform Riffle Shuffle

In this example we will analyze the uniform riffle shuffle, which is a more realistic model of how cards are shuffled by humans. The shuffle begins by dividing the deck in approximately two halves, and then merges the two halves in an approximately alternating manner. The reversed process, program **riffle** on Figure 4 which we analyze, takes a deck, samples a uniform random bit for each card, and then places all cards labeled with 0 on top of the deck without altering their relative order. After repeating this process k times, for every card i we have sampled a string of bits $(b_{i,0}, \dots, b_{i,k-1})$, and card i is on top of card j if, for some m, $b_{i,k} = b_{j,k}, b_{i,k-1} = b_{j,k-1}, \dots, b_{i,m} = b_{j,m}$ and $b_{i,m-1} < b_{j,m-1}$.

The vector b holds N bits, indexed by position; $\bar{b}$ negates each entry. We use shorthands for partitioning: $deck(b)$ and $deck(\bar{b})$ represent the sub-permutations from taking all positions where b is 0 and 1, respectively. Finally, cat concatenates two permutations.

We will take the coupling that always samples the same bit for the same card on both sides: $b(deck^{-1}(c))\langle 1\rangle = b(deck^{-1}(c))\langle 2\rangle$ for every $c \in C$. It is not hard to see that this coupling will eventually make the decks match. However, choosing an appropriate distance takes more care, since the Hamming distance may not always decrease under this coupling.

We define instead a semidistance (i.e., a function that satisfies all the distance axioms except for the triangle inequality) d_B in terms of a few concepts from the theory of permutations. We omit the fine details, which can be found in the full version. Assume we have a permutation $\pi\colon [N] \to [N]$ such that, for all $n \in [N]$, $deck_1[n] = deck_2[\pi(n)]$. We define a *block decomposition* of π to be a partition of the positions $B_1, \dots, B_j$ such that each block is contiguous, and π acts as a permutation on each B_i. A block decomposition is *minimal* if no block can be further decomposed; it is not hard to show that a minimal block decomposition must be unique. When $deck_1, deck_2$ are permutations (denoted $\mathsf{perm}(deck\langle 1\rangle, deck\langle 2\rangle)$), there exists a unique π as above, and we write $BD(deck_1, deck_2)$ for the block decomposition induced by two decks $deck_1$ and $deck_2$. We define now:

$$d_B(deck_1, deck_2) \triangleq \frac{1}{N^2} \sum_{c \in C} (|BD(deck_1, deck_2)(c)| - 1).$$

where $|BD(deck_1, deck_2)(c)|$ is the size of the block containing card c in $deck_1$ and $deck_2$; both positions must be in the same block. It is not hard to see that $|BD(deck_1, deck_2)(c)| = 0$ implies that c is at the same position in $deck_1$ and $deck_2$, (but the reverse implication may not hold) and that if d_B is 0, then the decks are equal. Now, we turn to the loop. Let Φ be the boolean assertion

$$\Phi \triangleq \mathsf{perm}(deck\langle 1\rangle, deck\langle 2\rangle) \wedge k\langle 1\rangle = k\langle 2\rangle \wedge (b \circ deck^{-1})\langle 1\rangle = (b \circ deck^{-1})\langle 2\rangle$$

By taking the following invariant expectation:

$$\mathcal{I} = [\neg\Phi] \cdot \infty + [\Phi] \cdot d_B \cdot (1/2)^{(K - k\langle 1\rangle)_+}$$

we can conclude

$$\widetilde{rpe}(\textbf{riffle}(deck, N, K), d_B) \le [\neg\Phi] + [\Phi] \cdot d_B \cdot (1/2)^K \le [\neg\Phi] + [\Phi] \cdot (1/2)^K$$

since the initial d_B is at most 1. Given that d_B assigns different decks a distance of at least $1/N^2$, by Theorem 2.5

$$v(K, N) = \max_{d_1, d_2 \in [N]} TV([\![\textbf{riffle}]\!](d_1, N, K), [\![\textbf{riffle}]\!](d_2, N, K)) \le N^2 \left(\frac{1}{2}\right)^K,$$

so the distributions converge to one another and to the uniform distribution exponentially quick. If we take $K \geq \log_2(N^2\rho)$, $v(K)$ is asymptotically bounded by $O(1/\rho)$ for large N. When setting $\rho = N$, we establish the following guarantee.

Theorem 6.4. *Let $K = 3\log N$, and $Perm([N])$ be the set of permutations over N. For any initial permutation of deck,*

$$TV(\mathbf{riffle}(deck, N, K), \mathbf{Unif}\{Perm([N])\}) \in O(1/N).$$

7 EXTENSIONS: PROVING UNIFORMITY

In Section 6, we showed that the Markov chains correspond to each example converge to a stationary distribution, but we did not shown that this distribution is the uniform distribution over states—if we had made an error in the implementation, the probabilistic program may converge to the wrong distribution. We can use our relational pre-expectation calculus along with Theorem 2.4 to show that the limit distribution is indeed uniform.

We illustrate the technique for the random-to-top shuffle, but the idea is applicable to all our examples. Consider any two permutations of the deck R_1, R_2, and the unary expectations

$$S_1(deck) \triangleq [deck = R_1] \qquad \text{and} \quad S_2(deck) \triangleq [deck = R_2].$$

To show that the shuffle converges to uniform, we need to show that the expected values of S_1 and S_2 converge to the same value. Recall that Theorem 2.4 states that for any initial states s_1, s_2,

$$\left|\mathbb{E}_{[\![\mathbf{rTop}]\!]s_1}[S_1] - \mathbb{E}_{[\![\mathbf{rTop}]\!]s_2}[S_2]\right| \leq |S_1 - S_2|^{\#}\left([\![\mathbf{rTop}]\!]s_1, [\![\mathbf{rTop}]\!]s_2\right)$$

so it suffices to show that the right hand side converges to zero.

Computing the relational pre-expectation of $|S_1 - S_2|$ directly is difficult, so we define an alternative distance. We can see R_1 and R_2 as defining a relation (actually, a permutation π over $[N]$) of pairs $(R_1[i], R_2[i])$ of cards that are at the same positions. We let d be the distance defined by:

$$d(deck\langle 1\rangle, deck\langle 2\rangle) \triangleq \sum_{i=0}^{N-1} [(deck\langle 1\rangle[i], deck\langle 2\rangle[i]) \notin \pi] \; .$$

We can show that $|S_1(deck\langle 1\rangle) - S_2(deck\langle 2\rangle)| \leq d(deck\langle 1\rangle, deck\langle 2\rangle)$, since d takes non-negative integer values, and whenever $d = 0$, then S_1 and S_2 can only be true simultaneously. So it suffices to show that the right-hand side converges to zero. This bound can also be established by our relational pre-expectation calculus in much the same way as in our proof for the random-to-top shuffle, but we use a different coupling. After sampling $p\langle 1\rangle$ on the first execution we just need to pick the $p\langle 2\rangle$ on the second such that $(deck\langle 1\rangle[p\langle 1\rangle], deck\langle 2\rangle[p\langle 2\rangle]) \in \pi$. This makes d decrease any time a new match is formed, and once a match is formed and moved to the top, it is never undone. By starting from the same permutation $deck\langle 1\rangle = deck\langle 2\rangle$, this analysis shows that the rate of convergence—this time to the *uniform* distribution—is the same as in our previous analysis of random-to-top: d converges to 0 at rate $(1 - 1/N)^K$.

8 EXTENSIONS: RULES FOR ASYNCHRONOUS REASONING

Our relational pre-expectation operator $\widetilde{rpe}(c, \mathcal{E})$ can often derive useful upper bounds on the Kantorovich distance $rpe(c, \mathcal{E})$, but it gives a trivial bound of infinity when the program c can take different branches on the two inputs. In this section, we develop techniques to give more useful bounds in the asynchronous case.

8.1 Asynchronous Rules for Bounding the Kantorovich Distance

Our asynchronous bounds will use one-sided relational operators $wpe\langle 1\rangle(c, \mathcal{E})$ (resp. $wpe\langle 2\rangle(c, \mathcal{E})$) that transform relational expectations by considering that the variables labeled with $\langle 2\rangle$ (resp. labeled with $\langle 1\rangle$) remain unchanged and then computing the unary weakest pre-expectation $wpe(c, \mathcal{E})$ as defined in Figure 5 (note in particular that $wpe\langle 1\rangle(x \leftarrow e, \mathcal{E})$ replaces only the variable labeled with $\langle 1\rangle$). We use the following soundness lemma for the left version of the operator, the one for the right version being analogous.

$$\begin{aligned}
wpe(\mathbf{skip}, \mathcal{E}) &\triangleq \mathcal{E} \\
wpe(x \leftarrow e, \mathcal{E}) &\triangleq \mathcal{E}\{e/x\} \\
wpe(x \xleftarrow{\$} d, \mathcal{E}) &\triangleq \lambda s.\mathbb{E}_{v\sim d}[\mathcal{E}\{v/x\}] \\
wpe(c; c', \mathcal{E}) &\triangleq wpe(c,\ wpe(c', \mathcal{E})) \\
wpe(\mathbf{if}\ e\ \mathbf{then}\ c\ \mathbf{else}\ c', \mathcal{E}) &\triangleq [e] \cdot wpe(c, \mathcal{E}) + [\neg e] \cdot wpe(c', \mathcal{E}) \\
wpe(\mathbf{while}\ e\ \mathbf{do}\ c, \mathcal{E}) &\triangleq \mathsf{lfp} X.[e] \cdot wpe(c, X) + [\neg e] \cdot \mathcal{E}
\end{aligned}$$

$$\begin{aligned}
wpe\langle 1\rangle(\mathbf{skip}, \mathcal{E}) &\triangleq \mathcal{E} \\
wpe\langle 1\rangle(x \leftarrow e, \mathcal{E}) &\triangleq \mathcal{E}\{e\langle 1\rangle/x\langle 1\rangle\} \\
wpe\langle 1\rangle(x \xleftarrow{\$} d, \mathcal{E}) &\triangleq \lambda s.\mathbb{E}_{v\sim d}[\mathcal{E}\{v/x\langle 1\rangle\}] \\
wpe\langle 1\rangle(c; c', \mathcal{E}) &\triangleq wpe\langle 1\rangle(c,\ wpe\langle 1\rangle(c', \mathcal{E})) \\
wpe\langle 1\rangle(\mathbf{if}\ e\ \mathbf{then}\ c\ \mathbf{else}\ c', \mathcal{E}) &\triangleq [e\langle 1\rangle] \cdot wpe\langle 1\rangle(c, \mathcal{E}) + [\neg e\langle 1\rangle] \cdot wpe\langle 1\rangle(c', \mathcal{E}) \\
wpe\langle 1\rangle(\mathbf{while}\ e\ \mathbf{do}\ c, \mathcal{E}) &\triangleq \mathsf{lfp} X.[e\langle 1\rangle] \cdot wpe\langle 1\rangle(c, X) + [\neg e\langle 1\rangle] \cdot \mathcal{E}
\end{aligned}$$

Fig. 5. Definition of the weakest pre-expectation operator $wpe(c, \mathcal{E})$ and the one-sided operator $wpe\langle 1\rangle(c, \mathcal{E})$

Lemma 8.1. *Let c be a pWhile program that is almost surely terminating, i.e., $wpe(c, 1) = 1$. Then, for all s_1, s_2, $\mathbb{E}_{s_1'\sim[\![c]\!]s_1}[\mathcal{E}(s_1', s_2)] \leq wpe\langle 1\rangle(c, \mathcal{E})(s_1, s_2)$.*

Now we can present our asynchronous rules for conditionals and loops:

Theorem 8.2. *Let c be a program that is almost surely terminating. Then:*

- *For conditionals with empty* **else** *branch, we can show:*

$$\begin{aligned}
rpe(\mathbf{if}\ e\ \mathbf{then}\ c, \mathcal{E}) \leq{}& [e\langle 1\rangle \wedge e\langle 2\rangle] \cdot \widetilde{rpe}(c, \mathcal{E}) + [e\langle 1\rangle \wedge \neg e\langle 2\rangle] \cdot wpe\langle 1\rangle(c, \mathcal{E}) \\
&+ [\neg e\langle 1\rangle \wedge e\langle 2\rangle] \cdot wpe\langle 2\rangle(c, \mathcal{E}) + [\neg e\langle 1\rangle \wedge \neg e\langle 2\rangle] \cdot \mathcal{E}
\end{aligned}$$

- *Let* **while** *e* **do** *c be an almost surely terminating loop, $\rho_i(s)$ be the probability that the loop does not terminate after executing the body at most i times starting from state s, and:*

$$M_i(\mathcal{E}, s_1, s_2) = \max\{\mathcal{E}(t_1, t_2) \mid t_1 \in supp([\![c_i]\!]s_1), t_2 \in supp([\![c_i]\!]s_2)\}$$

where c_i is the first i iterations of the loop. If ρ_i and M_i satisfy:

$$\lim_{i\to\infty} (\rho_i(s_1) + \rho_i(s_2)) \cdot M_i(\mathcal{E}, s_1, s_2) = 0$$

for any two states (s_1, s_2), and if $\mathcal{I}$ is an invariant satisfying

$$\begin{aligned}
&[e\langle 1\rangle \wedge e\langle 2\rangle] \cdot \widetilde{rpe}(c, \mathcal{I}) + [e\langle 1\rangle \wedge \neg e\langle 2\rangle] \cdot wpe\langle 1\rangle(c, \mathcal{I}) \\
&+ [\neg e\langle 1\rangle \wedge e\langle 2\rangle] \cdot wpe\langle 2\rangle(c, \mathcal{I}) + [\neg e\langle 1\rangle \wedge \neg e\langle 2\rangle] \cdot \mathcal{E} \leq \mathcal{I}\ ,
\end{aligned}$$

then $rpe(\mathbf{while}\ e\ \mathbf{do}\ c, \mathcal{E}) \leq \mathcal{I}$.

Proof Sketch. The soundness of the conditional rule follows a similar argument as soundness for the definition of $\widetilde{rpe}$ for conditionals, using Lemma 8.1 for the asynchronous cases. The soundness of the loop rule is more intricate, but it follows the same strategy as in Theorem 3.8: we define a loop characteristic function based on the conditional rule (now asynchronous), show that the least fixed-point lies above *rpe*—this step relies on the boundedness side-condition—and finally show that the invariant rule implies that $\mathcal{I}$ is a pre-fixed-point, so it must be above the fixed point. □

8.2 Example: Bounding the Distance between Binomial Distributions

Consider the following program, which simulates a binomial distribution:

```
binom($N$)
  $n \leftarrow 0$;
  $k \leftarrow 0$;
  while $n < N$ do
    $b \xleftarrow{\$} \mathbf{Bern}(p)$;
    if $b$ then $k \leftarrow k + 1$;
    $n \leftarrow n + 1$;
```

We treat $p \in [0, 1]$ as a fixed constant. We will compare the distribution on the output k starting from two inputs. Since the loops will run for different numbers of iterations if $N\langle 1\rangle \neq N\langle 2\rangle$, we will employ our asynchronous rule. We take the following invariant:

$$\mathcal{I} \triangleq |\, k\langle 1\rangle - k\langle 2\rangle + p \cdot (N\langle 1\rangle \ominus n\langle 1\rangle) - p \cdot (N\langle 2\rangle \ominus n\langle 2\rangle)\,| \;,$$

We will show the following invariant bound:

$$[(n <)\langle 1\rangle \wedge (n < N)\langle 2\rangle] \cdot \widetilde{rpe}(c, \mathcal{I}) + [(n < N)\langle 1\rangle \wedge (n \geq N)\langle 2\rangle] \cdot wpe\langle 1\rangle(c, \mathcal{I}) \\ + [(n \geq N)\langle 1\rangle \wedge (n < N)\langle 2\rangle] \cdot wpe\langle 2\rangle(c, \mathcal{I}) + [(n \geq N)\langle 1\rangle \wedge (n \geq N)\langle 2\rangle] \cdot \mathcal{E} \leq \mathcal{I} \;.$$

In the synchronous case, we can establish the invariant by applying Samp with the identity coupling; the inner conditional can also be analyzed synchronously. In the asynchronous case, computing the unary weakest pre-expectation establishes the invariant. Thus, the asynchronous loop rule (Theorem 8.2) gives:

$$rpe(w, |\, k\langle 1\rangle - k\langle 2\rangle\,|) \leq \mathcal{I}$$

where w is the loop. Applying the assignment rule, we have the bound:

$$rpe(\mathbf{binom}(N), |\, k\langle 1\rangle - k\langle 2\rangle\,|) \leq p \cdot |\, N\langle 1\rangle - N\langle 2\rangle\,|.$$

The side-condition of Theorem 2.4 holds, since for any initial state s, the loop terminates in at most $s(N)$ iterations. Thus, by Theorem 2.4, this bound implies that the expected values of the output k differ by at most $p \cdot |N\langle 1\rangle - N\langle 2\rangle|$ across the two runs.

9 RELATED WORK

Proving expected sensitivity of probabilistic programs. We have shown that the quantitative logic $\mathbb{E}$pRHL [Barthe et al. 2018] can be embedded into the framework of this paper (cf. Section 3.4), so we focus on other work. Wang et al. [2020] propose an alternative method based on martingales for proving the expected sensitivity of probabilistic programs. Their technique focuses on computing the expected sensitivity when the (expected) number of iterations for a loop may be different across two related executions (i.e., loops may be *asynchronous*); this is similar to our asynchronous rules from Section 8. However, Wang et al. [2020] also frame their target property in a slightly weaker way, showing that programs are Lipschitz continuous for *some* finite Lipschitz constant. In contrast, our method establishes bounds on this constant, which is an important aspect in many applications (e.g., it determines the rate of convergence for Markov chains). We are also able to handle the broader class of expected sensitivity properties arising from Kantorovich metrics, subsuming the notion considered by Wang et al. [2020] where the output distance is the absolute difference between two expected values.

Formal reasoning for probabilistic programs. Logics for probabilistic programs has been an active research area since the 1980s. Seminal work by Kozen [1985] defines a probabilistic propositional dynamic logic for reasoning about probabilistic programs, using real-valued functions rather than boolean assertions. Morgan et al. [1996] define a weakest pre-expectation calculus for a programming language with (demonic) non-determinism and probabilities. Extensions of this calculus with recursion, conditioning and signed expectations have been considered [Kaminski and Katoen 2017; Olmedo et al. 2018, 2016]. Kaminski et al. [2016] define a similar calculus for bounding expected run-times of probabilistic programs. These works do not prove relational properties of programs, and are unsuitable for verifying sensitivity.

Continuity in programs and process calculi. Formal reasoning about the continuity of deterministic programs has received some attention. Chaudhuri et al. [2010, 2012] were the first to give a sound, compositional framework for verifying that a program is continuous. Reed and Pierce [2010] gave a type system that can verify Lipschitz continuity of functional programs (see also [Azevedo de Amorim et al. 2014, 2017; Gaboardi et al. 2013; Winograd-Cort et al. 2017]). Recently, Huang et al. [2018] proposed the tool `PSense` which can perform sensitivity analysis of probabilistic programs. Their technique relies on symbolic computation using the symbolic verifier PSI and Mathematica, and supports, e.g., the Total Variation distance and the expectation distance. `PSense` cannot reason, however, about general Kantorovich distances, or unbounded loops.

Finally, in the process-algebra setting, compositional reasoning about metrics has received some attention. Gebler et al. [2016] used uniform continuity to reason about the distance between recursive processes in a compositional way, while Gebler and Tini [2018] recently defined specification formats that can check uniform continuity syntactically. syntactic manner. A more general framework for reasoning about metrics has been given by Bacci et al. [2018], who presented an algebraic axiomatization of Markov processes in quantitative equational logic. Their framework supports reasoning about various metrics, including the Kantorovich metric.

10 CONCLUSION

We defined a pre-expectation calculus to compute upper bounds for Kantorovich metrics, and applied it to prove convergence of reinforcement learning and card shuffling algorithms, algorithmic stability of SGD, and uniformity of limit distributions. Our calculus provides theoretical foundations for reasoning about quantitative relational properties of probabilistic programs.

There are several natural directions for future work. One possible extension is to lift the requirement that programs terminate with equal probability on pairs of executions, possibly by leveraging alternative notions of the Kantorovich metric that accommodate distributions of different weight [Piccoli and Rossi 2016]. Other directions include developing a relational version of quantitative separation logic [Batz et al. 2019], and use it for proving relational properties of probabilistic heap-manipulating programs.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their close reading and helpful suggestions. This research was partially supported by the National Science Foundation CCF and SaTC (1943130 and 2023222), Facebook, and the University of Wisconsin. Part of this research was conducted during the first author's visit to RWTH Aachen University.

REFERENCES

David Aldous. 1983. Random Walks on Finite Groups and Rapidly Mixing Markov Chains. In *Séminaire de Probabilités XVII 1981/82 (Lecture Notes in Mathematics, Vol. 986)*. Springer-Verlag, 243–297. https://eudml.org/doc/113445

Philip Amortila, Doina Precup, Prakash Panangaden, and Marc G. Bellemare. 2020. A Distributional Analysis of Sampling-Based Reinforcement Learning Algorithms. In *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS 2020, 26-28 August 2020, Online [Palermo, Sicily, Italy] (Proceedings of Machine Learning Research, Vol. 108)*, Silvia Chiappa and Roberto Calandra (Eds.). PMLR, 4357–4366. http://proceedings.mlr.press/v108/amortila20a.html

Robert B. Ash and Catherine A. Doleans-Dade. 2000. *Probability and Measure Theory.* Academic Press.

Arthur Azevedo de Amorim, Marco Gaboardi, Emilio Jesús Gallego Arias, and Justin Hsu. 2014. Really natural linear indexed type-checking. In *Symposium on Implementation and Application of Functional Programming Languages (IFL), Boston, Massachusetts.* ACM Press, 5:1–5:12. https://doi.org/10.1145/2746325.2746335

Arthur Azevedo de Amorim, Marco Gaboardi, Justin Hsu, Shin-ya Katsumata, and Ikram Cherigui. 2017. A semantic account of metric preservation. In *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Paris, France.* 545–556. https://doi.org/10.1145/3009837.3009890

Giorgio Bacci, Radu Mardare, Prakash Panangaden, and Gordon D. Plotkin. 2018. An Algebraic Theory of Markov Processes. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018, Oxford, UK, July 09-12, 2018*, Anuj Dawar and Erich Grädel (Eds.). ACM, 679–688. https://doi.org/10.1145/3209108.3209177

Gilles Barthe, Thomas Espitau, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. 2018. Proving expected sensitivity of probabilistic programs. *PACMPL* 2, POPL (2018), 57:1–57:29. https://doi.org/10.1145/3158145

Kevin Batz, Benjamin Lucien Kaminski, Joost-Pieter Katoen, Christoph Matheja, and Thomas Noll. 2019. Quantitative Separation Logic: A Logic for Reasoning About Probabilistic Pointer Programs. *PACMPL* 3, POPL (2019), 34:1–34:29. https://doi.org/10.1145/3290347

Kevin Batz, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Christoph Matheja. 2021. Relatively Complete Verification of Probabilistic Programs — An Expressive Language for Expectation-based Reasoning. *Proc. ACM Program. Lang.* 5, POPL (2021).

Nick Benton. 2004. Simple Relational Correctness Proofs for Static Analyses and Program Transformations. In *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Venice, Italy.* 14–25. https://doi.org/10.1145/964001.964003

Olivier Bousquet and André Elisseeff. 2002. Stability and Generalization. *Journal of Machine Learning Research* 2 (2002), 499–526. http://www.jmlr.org/papers/v2/bousquet02a.html

Swarat Chaudhuri, Sumit Gulwani, and Roberto Lublinerman. 2010. Continuity analysis of programs. In *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Madrid, Spain.* 57–70. https://doi.org/10.1145/1706299.1706308

Swarat Chaudhuri, Sumit Gulwani, and Roberto Lublinerman. 2012. Continuity and robustness of programs. *Commun. ACM* 55, 8 (2012), 107–115. https://doi.org/10.1145/2240236.2240262

Marco Gaboardi, Andreas Haeberlen, Justin Hsu, Arjun Narayan, and Benjamin C. Pierce. 2013. Linear dependent types for differential privacy. In *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Rome, Italy.* 357–370. https://doi.org/10.1145/2429069.2429113

Daniel Gebler, Kim G. Larsen, and Simone Tini. 2016. Compositional bisimulation metric reasoning with probabilistic process calculi. *Logical Methods in Computer Science* 12, 4 (2016). https://doi.org/10.2168/LMCS-12(4:12)2016

Daniel Gebler and Simone Tini. 2018. SOS specifications for uniformly continuous operators. *J. Comput. Syst. Sci.* 92 (2018), 113–151. https://doi.org/10.1016/j.jcss.2017.09.011

Friedrich Gretz, Joost-Pieter Katoen, and Annabelle McIver. 2014. Operational versus weakest pre-expectation semantics for the probabilistic guarded command language. *Perform. Evaluation* 73 (2014), 110–132. https://doi.org/10.1016/j.peva.2013.11.004

Moritz Hardt, Ben Recht, and Yoram Singer. 2016. Train faster, generalize better: Stability of stochastic gradient descent. In *International Conference on Machine Learning (ICML), New York, NY (Journal of Machine Learning Research, Vol. 48).* JMLR.org, 1225–1234. http://jmlr.org/proceedings/papers/v48/hardt16.html

Zixin Huang, Zhenbang Wang, and Sasa Misailovic. 2018. PSense: Automatic Sensitivity Analysis for Probabilistic Programs. In *Automated Technology for Verification and Analysis - 16th International Symposium, ATVA 2018, Los Angeles, CA, USA, October 7-10, 2018, Proceedings (LNCS, Vol. 11138)*, Shuvendu K. Lahiri and Chao Wang (Eds.). Springer, 387–403. https://doi.org/10.1007/978-3-030-01090-4_23

Benjamin Lucien Kaminski and Joost-Pieter Katoen. 2017. A weakest pre-expectation semantics for mixed-sign expectations. In *LICS.* IEEE Computer Society, 1–12. https://doi.org/10.1109/LICS.2017.8005153

Benjamin Lucien Kaminski, Joost-Pieter Katoen, Christoph Matheja, and Federico Olmedo. 2016. Weakest Precondition Reasoning for Expected Run-Times of Probabilistic Programs. In *European Symposium on Programming (ESOP), Eindhoven, The Netherlands (Lecture Notes in Computer Science, Vol. 9632).* Springer-Verlag, 364–389. https://doi.org/10.1007/978-3-

662-49498-1_15

Dexter Kozen. 1985. A Probabilistic PDL. *J. Comput. System Sci.* 30, 2 (1985), 162–178. https://doi.org/10.1016/0022-0000(85)90012-1

Annabelle McIver and Carroll Morgan. 2005. *Abstraction, Refinement and Proof for Probabilistic Systems.* Springer.

Carroll Morgan, Annabelle McIver, and Karen Seidel. 1996. Probabilistic Predicate Transformers. *ACM Transactions on Programming Languages and Systems* 18, 3 (1996), 325–353.

Federico Olmedo, Friedrich Gretz, Nils Jansen, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Annabelle McIver. 2018. Conditioning in Probabilistic Programming. *ACM Trans. Program. Lang. Syst.* 40, 1 (2018), 4:1–4:50. https://doi.org/10.1145/3156018

Federico Olmedo, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Christoph Matheja. 2016. Reasoning about Recursive Probabilistic Programs. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, Martin Grohe, Eric Koskinen, and Natarajan Shankar (Eds.). ACM, 672–681. https://doi.org/10.1145/2933575.2935317

David Park. 1969. Fixpoint Induction and Proofs of Program Properties. *Machine Intelligence* 5 (1969).

Benedetto Piccoli and Francesco Rossi. 2016. On Properties of the Generalized Wasserstein Distance. *Archive for Rational Mechanics and Analysis* 222, 3 (01 Dec 2016), 1339–1365. https://doi.org/10.1007/s00205-016-1026-7

Jason Reed and Benjamin C Pierce. 2010. Distance Makes the Types Grow Stronger: A Calculus for Differential Privacy. In *ACM SIGPLAN International Conference on Functional Programming (ICFP), Baltimore, Maryland.* https://doi.org/10.1145/1863543.1863568

Richard S. Sutton. 1988. Learning to Predict by the Methods of Temporal Differences. *Mach. Learn.* 3 (1988), 9–44. https://doi.org/10.1007/BF00115009

Cédric Villani. 2008. *Optimal Transport: Old and New.* Springer-Verlag.

Peixin Wang, Hongfei Fu, Krishnendu Chatterjee, Yuxin Deng, and Ming Xu. 2020. Proving expected sensitivity of probabilistic programs with randomized variable-dependent termination time. *Proc. ACM Program. Lang.* 4, POPL (2020), 25:1–25:30. https://doi.org/10.1145/3371093

Daniel Winograd-Cort, Andreas Haeberlen, Aaron Roth, and Benjamin C. Pierce. 2017. A framework for adaptive differential privacy. In *ACM SIGPLAN International Conference on Functional Programming (ICFP), Oxford, England.* 10:1–10:29. https://doi.org/10.1145/3110254