PROBABILISTIC COUPLINGS FOR PROBABILISTIC REASONING

Justin Hsu

A DISSERTATION

in

Computer and Information Science

Presented to the Faculties of the University of Pennsylvania in Partial
Fulfillment of the Requirements for the Degree of Doctor of Philosophy

2017

Dissertation Supervisors
Benjamin C. Pierce
Aaron Roth

Graduate Group Chairperson
Lyle Ungar

Dissertation Committee
Gilles Barthe
Sampath Kannan
Val Tannen
Steve Zdancewic

For my family,
for my teachers.

Acknowledgments

It takes a village, as they say. I am extremely fortunate to have two amazing advisers: Benjamin Pierce, who taught me to ponder slowly, and Aaron Roth, who taught me to think rapidly. This thesis would not have existed without Gilles Barthe, serving as a third, unofficial adviser. Gilles hosted me during two highly productive summers at the IMDEA Software Institute in Madrid and remains an inspiring (and boundless) source of energy and enthusiasm. He and his longtime collaborators, Pierre-Yves Strub and Benjamin Grégoire, provided a wealth of technical expertise and a theoretical setting that turned out to be far richer than anyone had thought.

The strong theory and formal methods faculty at Penn, including Sanjeev Khanna, Stephanie Weirich, Michael Kearns, Steve Zdancewic, Sampath Kannan, Val Tannen, Sudipto Guha, and Rajeev Alur, have taught me more than I can hope to remember. I am also indebted to Marco Gaboardi and Emilio Jesús Gallego Arias for launching me to the research frontiers. Other fellow students and postdocs, including Arthur Azevedo de Amorim, Rachel Cummings, Michael Greenberg, Radoslav Ivanov, Shahin Jabbari, Matthew Joseph, Jamie Morgenstern, Christine Rizkallah, Ryan Rogers, Nikos Vasilakis, and Steven Wu, kept morale at a good level, to say the least.

At Stanford, Greg Lavender's one-off course on Haskell gave me my first glimpse of programming languages as an intellectual field. Without that formative experience, I never would have gone on to graduate school in computer science. Finally, I am deeply grateful for the constant support and encouragement from my parents Joyce and David, my sister Tammy, and the rest of my family.

<div align="right">

London, UK
November 2, 2017

</div>

Abstract

This thesis explores *proofs by coupling* from the perspective of formal verification. Long employed in probability theory and theoretical computer science, these proofs construct *couplings* between the output distributions of two probabilistic processes. Couplings can imply various *probabilistic relational properties*, guarantees that compare two runs of a probabilistic computation.

To give a formal account of this clean proof technique, we first show that proofs in the program logic PRHL (probabilistic Relational Hoare Logic) describe couplings. We formalize couplings that establish various probabilistic properties, including distribution equivalence, convergence, and stochastic domination. Then we deepen the connection between couplings and PRHL by giving a proofs-as-programs interpretation: a coupling proof encodes a probabilistic *product program*, whose properties imply relational properties of the original two programs. We design the logic ×PRHL (*product* PRHL) to build the product program, with extensions to model more advanced constructions including *shift coupling* and *path coupling*.

We then develop an approximate version of probabilistic coupling, based on *approximate liftings*. It is known that the existence of an approximate lifting implies *differential privacy*, a relational notion of statistical privacy. We propose a corresponding proof technique—*proof by approximate coupling*—inspired by the logic APRHL, a version of PRHL for building approximate liftings. Drawing on ideas from existing privacy proofs, we extend APRHL with novel proof rules for constructing new approximate couplings. We give approximate coupling proofs of privacy for the *Report-noisy-max* and *Sparse Vector* mechanisms, well-known algorithms from the privacy literature with notoriously subtle privacy proofs, and produce the first formalized proof of privacy for these algorithms in APRHL.

Finally, we enrich the theory of approximate couplings with several more sophisticated constructions: a principle for showing accuracy-dependent privacy, a generalization of the advanced composition theorem from differential privacy, and an optimal approximate coupling relating two subsets of samples. We also show equivalences between approximate couplings and other existing definitions. These ingredients support the first formalized proof of privacy for the *Between Thresholds* mechanism, an extension of the Sparse Vector mechanism.

Contents

List of Figures

Chapter 1

Introduction

Randomized algorithms have long stimulated the imagination of computer scientists. Endowed with the power to draw random samples, these algorithms provide sophisticated guarantees far beyond the reach of deterministic computations. However, their proofs of correctness are often highly intricate, employing specialized techniques to reason about randomness.

This thesis investigates one such tool—*probabilistic coupling*—for proving *probabilistic relational properties*, which compare executions of two randomized algorithms. Couplings are a familiar concept in probability theory and theoretical computer science, where they support a proof technique called *proof by coupling*. We explore the reasoning principle behind these proofs, identifying their theoretical underpinnings, clarifying their structure, and enabling formal verification.

1.1   Challenges in probabilistic reasoning

While probabilistic programs aren't much harder to express than their deterministic counterparts, they are significantly more challenging to reason about. To see why, suppose we want to prove a property about the output of an algorithm for all inputs. In a deterministic algorithm, each concrete input produces a single trace through the program. Since different paths correspond to distinct inputs, we can freely group similar traces together and reason about each group on its own. The code of the algorithm naturally guides the proof: at a branching instruction, for instance, we may classify the executions according to the path they take and then consider each behavior separately. In this way, we can reason about a complex program by focusing on simpler cases.

For randomized algorithms, this neat picture is considerably more complicated. A single execution now comprises multiple traces, each with its own probability. Relations between trace probabilities make it difficult to reason about paths separately. At a conditional statement, for instance, the execution has some probability of taking the first branch and some probability of taking the second branch; in a sense, the computation takes *both* branches. If we reason about these two cases in isolation, we must track the probabilities of each branch in order to join the cases when the paths later merge. This is challenging even for small programs, as a path's probability can have complex dependencies on the input and on the probabilities of other possible traces. If we instead reason about both behaviors together, we must provide a single analysis for executions that behave quite differently.

Broadly speaking, then, a central challenge of probabilistic reasoning is to organize the various execution behaviors into manageable cases while cleanly tracking the relationship across different groups. To tackle this problem, researchers in randomized algorithms have crafted a rich array of conceptual tools to construct their proofs, simplifying arguments by cleverly abstracting away uninteresting technical details. Also known as *proof techniques*, these instruments can be sophisticated and highly specialized—often tailored to a single property, as a kind of logical scalpel—but the most useful ones strike a fine balance: specific enough to pare logical arguments down to just their essential points, general enough to support proofs for a broad class of properties. A proof

technique is a reusable component for analyzing algorithms, and is as much of an intellectual contribution as any new proof or algorithm.

1.2 Couplings and relational properties

In this thesis we explore a proof technique for *probabilistic relational properties*, guarantees comparing the runs of two randomized algorithms. Such properties are commonplace in computer science and probability theory. Examples include:

- **Probabilistic equivalence**: two probabilistic programs produce equal distributions.

- **Stochastic domination**: one probabilistic program is more likely than another to produce large outputs.

- **Convergence** (also **mixing**): the output distributions of two probabilistic loops approach each other as the loops execute more iterations.

- **Indistinguishability** (also **differential privacy**): the output distributions of two probabilistic programs are close together. For instance, *differential privacy* requires that two similar inputs—say, the real private database and a hypothetical version with one individual's data omitted—yield similar output distributions.

- **Truthfulness** (also **Nash equilibrium**): an agent's average utility is larger when reporting an honest value instead of deviating to a misleading value.

At first glance, relational properties appear to be even harder to establish than standard, non-relational properties—instead of analyzing a single probabilistic computation, we now need to deal with two. (Indeed, any property of a single program can be viewed as a relational property between the target program and the trivial, do-nothing program.) However, relational properties often relate two highly similar programs, even comparing the same program on two possible inputs. In these cases, we can leverage a powerful abstraction and an associated proof technique from probability theory—*probabilistic coupling* and *proof by coupling*.

The fundamental observation is that probabilistic relational properties compare computations in two different worlds, assuming no particular correlation between random samples. Accordingly, we may freely assume any correlation we like for the purposes of the proof—a relational property holds (or doesn't hold) regardless of which one we pick. For instance, if two programs generate identical output distributions, this holds whether they share coin flips or take independent samples; relational properties don't require that the two programs use separate randomness. By carefully arranging the correlation, we can reason about two executions as if they were linked in some convenient way.

To take advantage of this freedom, we need some way to design specific correlations between program executions. In principle, this can be a highly challenging task. The two runs may take samples from different distributions, and it is unclear exactly how they can or should share randomness. When the two programs have similar shapes, however, we can link two computations in a step-by-step fashion. First, correlations between intermediate samples can be described by *probabilistic couplings*, joint distributions over pairs. For example, a valid coupling of two fair coin flips could specify that the draws take opposite values; the correlated distribution would produce "(heads, tails)" and "(tails, heads)" with equal probability. A coupling formalizes what it means to share randomness: a single source of randomness simulates draws from two distributions. Since randomness can be shared in different ways, two distributions typically support a variety of distinct couplings.

A *proof by coupling*, then, describes two correlated executions by piecing together couplings for corresponding pairs of sampling instructions. In the course of a proof, we can imagine stepping through the two programs in parallel, selecting couplings along the way. For instance, if we apply the opposite coupling to link a coin flip in one program with a coin flip in the other, we may assume the samples remain opposite when analyzing the rest of the programs. By flowing these relations forward from two initial inputs, a proof by coupling can focus on just

pairs of similar executions as it builds up to a coupling between two output distributions. This is the main product of the proof: features of the final coupling imply properties about the output distributions, and hence relational properties about the original programs.

Working in tandem, couplings and proofs by couplings can simplify probabilistic reasoning in several ways.

- **Reduce to one source of randomness.** By analyzing two runs as if they shared a single source of randomness, we can reason about two programs as if they were one.

- **Abstract away probabilities.** Proofs by coupling isolate probabilistic reasoning from the non-probabilistic parts of the proof, which are more straightforward. We only need to think about probabilistic aspects when we select couplings at the sampling instructions; throughout the rest of the programs, we can reason purely in terms of deterministic relations between the two runs.

- **Enable compositional, structured reasoning.** By focusing on each step of an algorithm individually and then smoothly combining the results, the coupling proof technique enables a highly modular style of reasoning guided by the code of the program.

Proofs by coupling are also surprisingly flexible; many probabilistic relational properties, including the examples listed above, can be proved in this way. Individual couplings can also be combined in various subtle ways, giving rise to a rich diversity of coupling proofs.

1.3   A formal study of proofs by coupling

While couplings proofs originate from probability theory as a tool for human reasoning, *formal verification* will be the setting for our investigation. Our perspective affords two distinct advantages.

- The theory of formal verification provides a wealth of concepts to precisely describe and analyze proof systems. By studying coupling proofs in these terms, we can give a fresh understanding of this classical proof technique. As a consequence, we can extend proofs by coupling to target new guarantees, unifying seemingly unrelated properties and simplifying their proofs.

- Formal verification systems provide a natural domain to apply our insights. First, couplings enable clean proofs for properties that are traditionally challenging for computers to verify. Existing techniques can also be considered in a new light, clarifying why certain features are useful and revealing possible enhancements.

The technical chapters of this thesis fall into two parts. Chapters 2 and 3 concern probabilistic couplings, while Chapters 4 and 5 investigate approximate couplings. General themes and intuitions developed in the first half influence the second half, but the two parts are largely self-contained and can be read independently.

Chapter 2 begins our study of probabilistic couplings in formal verification. We observe that the program logic PRHL (*probabilistic Relational Hoare Logic*), originally proposed by Barthe, Grégoire, and Zanella-Béguelin (2009) for verifying proofs of cryptographic security, is in fact a logic for formally constructing probabilistic couplings. Using this connection, we formalize classical coupling proofs establishing equivalence, convergence, and stochastic domination of probabilistic processes.

Chapter 3 deepens our correspondence between couplings and PRHL. First, coupling proofs describe how to meld two probabilistic programs into a single program; in formal verification, such a construction is known as a *product program*. Accordingly, we show that PRHL proofs encode a novel kind of product program called the *coupled product*, reflecting the structure of a coupling proof. This idea recalls a central theme in logic and computer science: formal proofs can be interpreted as computations, a so-called *proofs-as-programs* or *Curry-Howard* correspondence. Concretely, we extend PRHL to a logic ×PRHL (*product* PRHL) that constructs the product program alongside the coupling proof. Then, we design a new loop rule inspired by *shift coupling*, a way to build couplings asynchronously.

As applications, we formalize rapid mixing for several Markov chains. Our approach can also capture a simplified version of the *path coupling* technique introduced by Bubley and Dyer (1997).

Chapter 4 turns our focus to a generalization of couplings: *approximate couplings*. These couplings are closely related to *differential privacy*, a quantitative, relational property modeling statistical privacy. We begin with a candidate definition of approximate coupling, refining several existing notions. We then reverse-engineer a corresponding proof technique called *proof by approximate coupling* from the program logic APRHL, an approximate version of PRHL proposed by Barthe, Köpf, Olmedo, and Zanella-Béguelin (2013c). Taking inspiration from this proof technique, we show how two new approximate couplings of the Laplace distribution and a construction called *pointwise equality* enable an approximate coupling proof of privacy for the *Report-noisy-max* and *Sparse Vector* mechanisms. Our proofs are simpler than existing proofs—which were notoriously difficult to get right (Lyu, Su, and Li, 2017)—and extend to natural variants of the algorithms. We realize our proof in an extension of APRHL, arriving at the first formalized privacy proofs for these mechanisms.

Chapter 5 presents a handful of advanced constructions for approximate couplings: (i) a principle for proving accuracy-dependent privacy; (ii) a construction for linking two subsets of samples; and (iii) a composition principle generalizing the advanced composition theorem from differential privacy. We also clarify the landscape of existing definitions by proving equivalences between approximate couplings and prior notions of approximate equivalence. Combining these ingredients, we give a proof by approximate coupling establishing differential privacy for the *Between Thresholds* mechanism by Bun, Steinke, and Ullman (2017). After extending APRHL with several rules corresponding to our constructions, we achieve the first formalized privacy proof for this algorithm.

Chapter 6 surveys concurrent work on couplings and formal verification, outlining promising directions for further developing the theory and application of proofs by coupling.

A note about mechanical verification.   The gold standard in formal verification is *mechanized proof*, where every step has been fully computer-checked. The logics we will develop are highly suitable for computer verification, due to their highly structured proofs, but we do not mechanically verify coupling proofs as part of this thesis. Instead, we will describe *formalized proofs* in the logic on paper. Prototype implementations in the EASYCRYPT framework (Barthe, Dupressoir, Grégoire, Kunz, Schmidt, and Strub, 2013b) can machine-check versions of the coupling proofs we will see (see, e.g., Barthe et al. (2013c) and Buch (2017)), but the current implementations are not precisely aligned with our logics.

Chapter 2

Couplings à la formal verification

To begin our formal investigation of coupling proofs, we first provide the necessary mathematical background (Section 2.1), and then draw a deep connection between coupling proofs and the program logic PRHL (Section 2.2); this observation is the principal conceptual contribution of this chapter and forms the foundation for the entire thesis. We show how to formalize several examples of couplings (Section 2.3), and discuss related work on relational program logics and probabilistic liftings (Section 2.4).

2.1 Mathematical preliminaries

A discrete probability distribution associates each element of a set with a number in $[0, 1]$, representing its *probability*. In order to model programs that may not terminate, we work with a slightly more general notion called a *sub-distribution*.

Definition 2.1.1. A (discrete) *sub-distribution* over a countable set $\mathcal{A}$ is a map $\mu : \mathcal{A} \to [0, 1]$ taking each element of $\mathcal{A}$ to a numeric weight such that the weights sum to at most 1:

$$\sum_{a \in \mathcal{A}} \mu(a) \leq 1.$$

We write **SDistr**$(\mathcal{A})$ for the set of all sub-distributions over $\mathcal{A}$. When the weights sum to 1, we call μ a *proper distribution*; we write **Distr**$(\mathcal{A})$ for the set of all proper distributions over $\mathcal{A}$. The *empty* or *null sub-distribution* $\bot$ assigns weight 0 to all elements.

We work with discrete sub-distributions throughout. While this is certainly a restriction—excluding, for instance, standard distributions over the real numbers—many interesting coupling proofs can already be expressed in our setting. Where necessary, we will use discrete versions of standard, continuous distributions. Our results should mostly carry over to the continuous setting, as couplings are frequently used on continuous distributions in probability theory, but the general case introduces measure-theoretic technicalities (e.g., working with integrals rather than sums, checking sets are measurable, etc.) that would distract from our primary focus. We discuss this issue further in Chapter 6.

We need several concepts and notations related to discrete distributions. First, the probability of a set $\mathcal{S} \subseteq \mathcal{A}$:

$$\mu(\mathcal{S}) \triangleq \sum_{a \in \mathcal{S}} \mu(a).$$

The *support* of a sub-distribution is the set of elements with positive probability:

$$\mathrm{supp}(\mu) \triangleq \{a \in \mathcal{A} \mid \mu(a) > 0\}.$$

The *weight* of a sub-distribution is the total probability of all elements:

$$|\mu| \triangleq \sum_{a \in \mathcal{A}} \mu(a).$$

Sub-distributions can be ordered pointwise: $\mu_1 \leq \mu_2$ if $\mu_1(a) \leq \mu_2(a)$ for every element $a \in \mathcal{A}$. Finally, given a function $f : \mathcal{A} \to \mathcal{B}$ where $\mathcal{B}$ is numeric (like the integers or the reals), its *expected value* over a sub-distribution μ is

$$\mathbb{E}_{\mu}[f] \triangleq \mathbb{E}_{a \sim \mu}[f(a)] \triangleq \sum_{a \in \mathcal{A}} f(a) \cdot \mu(a).$$

Under light assumptions, the expected value is guaranteed to exist (for instance, when f is a bounded function).

To transform sub-distributions, we can lift a function $f : \mathcal{A} \to \mathcal{B}$ on sets to a map $f^{\sharp} : \mathbf{SDistr}(\mathcal{A}) \to \mathbf{SDistr}(\mathcal{B})$ via $f^{\sharp}(\mu)(b) \triangleq \mu(f^{-1}(b))$. For example, let $p_1 : \mathcal{A}_1 \times \mathcal{A}_2 \to \mathcal{A}_1$ and $p_2 : \mathcal{A}_1 \times \mathcal{A}_2 \to \mathcal{A}_2$ be the first and second projections from a pair. The corresponding *probabilistic projections* $\pi_1 : \mathbf{SDistr}(\mathcal{A}_1 \times \mathcal{A}_2) \to \mathbf{SDistr}(\mathcal{A}_1)$ and $\pi_2 : \mathbf{SDistr}(\mathcal{A}_1 \times \mathcal{A}_2) \to \mathbf{SDistr}(\mathcal{A}_2)$ are defined by

$$\pi_1(\mu)(a_1) \triangleq p_1^{\sharp}(\mu)(a_1) = \sum_{a_2 \in \mathcal{A}_2} \mu(a_1, a_2)$$

$$\pi_2(\mu)(a_2) \triangleq p_2^{\sharp}(\mu)(a_2) = \sum_{a_1 \in \mathcal{A}_1} \mu(a_1, a_2).$$

We call a sub-distribution μ over pairs a *joint sub-distribution*, and the projected sub-distributions $\pi_1(\mu)$ and $\pi_2(\mu)$ the *first* and *second marginals*, respectively.

Probabilistic couplings and liftings

A probabilistic coupling models two distributions with a single joint distribution.

Definition 2.1.2. Given μ_1, μ_2 sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$, a sub-distribution μ over pairs $\mathcal{A}_1 \times \mathcal{A}_2$ is a *coupling* for (μ_1, μ_2) if $\pi_1(\mu) = \mu_1$ and $\pi_2(\mu) = \mu_2$.

Generally, couplings are not unique—different witnesses represent different ways to share randomness between two distributions. To give a few examples, we first introduce some standard distributions.

Definition 2.1.3. Let $\mathcal{A}$ be a finite, non-empty set. The *uniform distribution* over $\mathcal{A}$, written $\mathbf{Unif}(\mathcal{A})$, assigns probability $1/|\mathcal{A}|$ to each element. We write $\mathbf{Flip}$ for the uniform distribution over booleans, the distribution of a fair coin flip.

Example 2.1.4 (Couplings from bijections)**.** We can give two distinct couplings of $(\mathbf{Flip}, \mathbf{Flip})$:

Identity coupling:

$$\mu_{\mathrm{id}}(a_1, a_2) \triangleq \begin{cases} 1/2 & : a_1 = a_2 \\ 0 & : \text{otherwise.} \end{cases}$$

Negation coupling:

$$\mu_{\neg}(a_1, a_2) \triangleq \begin{cases} 1/2 & : \neg a_1 = a_2 \\ 0 & : \text{otherwise.} \end{cases}$$

More generally, any bijection $f : \mathcal{A} \to \mathcal{A}$ yields a coupling of $(\mathbf{Unif}(\mathcal{A}), \mathbf{Unif}(\mathcal{A}))$:

$$\mu_f(a_1, a_2) \triangleq \begin{cases} 1/|\mathcal{A}| & : f(a_1) = a_2 \\ 0 & : \text{otherwise.} \end{cases}$$

This coupling matches samples: each sample a from the first distribution is paired with a corresponding sample $f(a)$ from the second distribution. To take two correlated samples from this coupling, we can imagine first sampling from the first distribution, and then applying f to produce a sample for the second distribution. When f is a bijection, this gives a valid coupling for two uniform distributions: viewed separately, both the first and second correlated samples are distributed uniformly.

For more general distributions, if a_1 and a_2 have different probabilities under μ_1 and μ_2 then the correlated distribution cannot return $(a_1, -)$ and $(-, a_2)$ with equal probabilities; for instance, a bijection with $f(a_1) = a_2$ would not give a valid coupling. However, general distributions can be coupled in other ways.

Example 2.1.5. Let μ be a sub-distribution over $\mathcal{A}$. The *identity coupling* of (μ, μ) is

$$\mu_{\mathrm{id}}(a_1, a_2) \triangleq \begin{cases} \mu(a) & : a_1 = a_2 = a \\ 0 & : \text{otherwise.} \end{cases}$$

Sampling from this coupling yields a pair of equal values.

Example 2.1.6. Let μ_1, μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$. The *independent* or *trivial* coupling is

$$\mu_\times(a_1, a_2) \triangleq \mu_1(a_1) \cdot \mu_2(a_2).$$

This coupling models μ_1 and μ_2 as independent distributions: sampling from this coupling is equivalent to first sampling from μ_1 and then pairing with a fresh draw from μ_2. The coupled distributions must be proper in order to ensure the marginal conditions.

Since any two proper distributions can be coupled by the trivial coupling, the mere existence of a coupling yields little information. Couplings are more useful when the joint distribution satisfies additional conditions, for instance when all elements in the support satisfy some property.

Definition 2.1.7 (Lifting). Let μ_1, μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$, and let $\mathcal{R} \subseteq \mathcal{A}_1 \times \mathcal{A}_2$ be a relation. A sub-distribution μ over pairs $\mathcal{A}_1 \times \mathcal{A}_2$ is a *witness* for the $\mathcal{R}$-*lifting* of (μ_1, μ_2) if:

1. μ is a coupling for (μ_1, μ_2), and

2. $\mathrm{supp}(\mu) \subseteq \mathcal{R}$.

If there exists μ satisfying these two conditions, we say μ_1 and μ_2 are related by the *lifting of $\mathcal{R}$* and write

$$\mu_1 \, \mathcal{R}^\sharp \, \mu_2.$$

We typically express $\mathcal{R}$ using set notation, i.e.,

$$\mathcal{R} = \{(a_1, a_2) \in \mathcal{A}_1 \times \mathcal{A}_2 \mid \Phi(a_1, a_2)\}$$

where Φ is a logical formula. When Φ is a standard mathematical relation (e.g., equality), we leave $\mathcal{A}_1$ and $\mathcal{A}_2$ implicit and just write Φ, sometimes enclosed by parentheses (Φ) for clarity.

Example 2.1.8. Many of the couplings we saw before are more precisely described as liftings.

Bijection coupling. For a bijection $f : \mathcal{A} \to \mathcal{A}$, the coupling in Example 2.1.4 witnesses the lifting

$$\mathbf{Unif}(\mathcal{A}) \, \{(a_1, a_2) \mid f(a_1) = a_2\}^\sharp \, \mathbf{Unif}(\mathcal{A}).$$

Identity coupling. The coupling in Example 2.1.5 witnesses the lifting

$$\mu \, (=)^\sharp \, \mu.$$

Trivial coupling. The coupling in Example 2.1.6 witnesses the lifting

$$\mu_1 \; \top^\sharp \; \mu_2.$$

($\top \triangleq \mathcal{A}_1 \times \mathcal{A}_2$ is the trivial relation relating all pairs of elements.)

Liftings were originally introduced in research on *probabilistic bisimulation*, a technique for verifying equivalence of two probabilistic transition systems. By viewing liftings as a particular kind of coupling, we can repurpose verification tools to prove new properties by constructing couplings, while leveraging ideas from the coupling literature to enrich existing systems. Before we get to that, let's see how the existence of a coupling can imply useful probabilistic properties.

Useful consequences of couplings and liftings

If there exists a coupling μ between (μ_1, μ_2) satisfying certain properties, we can deduce probabilistic properties about μ_1 and μ_2. First of all, two coupled distributions have equal weight.

Proposition 2.1.9 (Equality of weight). *Suppose μ_1 and μ_2 are sub-distributions over $\mathcal{A}$ such that there exists a coupling μ of μ_1 and μ_2. Then $|\mu_1| = |\mu_2|$.*

This follows because μ_1 and μ_2 are both projections of μ, and projections preserve weight. Couplings can also show that two distributions are equal.

Proposition 2.1.10 (Equality of distributions). *Suppose μ_1 and μ_2 are sub-distributions over $\mathcal{A}$. Then $\mu_1 = \mu_2$ if and only if there is a lifting $\mu_1 \; (=)^\sharp \; \mu_2$.*

Proof. For the forward direction, define $\mu(a, a) \triangleq \mu_1(a) = \mu_2(a)$ and $\mu(a_1, a_2) \triangleq 0$ otherwise. Evidently, μ has support in the equality relation $(=)$ and also has the desired marginals: $\pi_1(\mu) = \mu_1$ and $\pi_2(\mu) = \mu_2$. Thus μ is a witness to the desired lifting.

For the reverse direction, let the witness be μ. By the support condition, $\pi_1(\mu)(a) = \pi_2(\mu)(a)$ for every $a \in \mathcal{A}$. Since the left and right sides are equal to $\mu_1(a)$ and $\mu_2(a)$ respectively by the marginal conditions, $\mu_1(a) = \mu_2(a)$ for every a. So, μ_1 and μ_2 are equal. $\square$

In some cases we can show results in the converse direction: if a property of two distributions holds, then there exists a particular lifting. To give some examples, we first introduce a powerful equivalence due to Strassen (1965).

Theorem 2.1.11. *Let μ_1, μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$, and let $\mathcal{R}$ be a binary relation over $\mathcal{A}_1$ and $\mathcal{A}_2$. Then the lifting $\mu_1 \; \mathcal{R}^\sharp \; \mu_2$ implies $\mu_1(\mathcal{S}_1) \leq \mu_2(\mathcal{R}(\mathcal{S}_1))$ for every subset $\mathcal{S}_1 \subseteq \mathcal{A}_1$, where $\mathcal{R}(\mathcal{S}_1) \subseteq \mathcal{A}_2$ is the* image *of $\mathcal{S}_1$ under $\mathcal{R}$:*

$$\mathcal{R}(\mathcal{S}_1) \triangleq \{a_2 \in \mathcal{A}_2 \mid \exists a_1 \in \mathcal{A}_1, \; (a_1, a_2) \in \mathcal{R}\}.$$

(For instance, if $\mathcal{A}_1 = \mathcal{A}_2 = \mathbb{N}$ and $\mathcal{R}$ is the relation $\leq$, then $\mathcal{R}(\mathcal{S})$ is the set of all natural numbers larger than $\min \mathcal{S}$.) The converse holds if μ_1 and μ_2 have equal weight.

Strassen proved Theorem 2.1.11 for continuous (proper) distributions using deep results from probability theory. In our discrete setting, there is an elementary proof by the maximum flow-minimum cut theorem; the proof also establishes a mild generalization to sub-distributions. So as not to interrupt the flow here, we defer details of the proof to Chapter 5. For now, we use this theorem to illustrate a few more useful consequences of liftings. For starters, couplings can bound the probability of an event in the first distribution by the probability of an event in the second distribution.

Proposition 2.1.12. *Suppose μ_1, μ_2 are sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$ respectively, and consider two subsets $\mathcal{S}_1 \subseteq \mathcal{A}_1$ and $\mathcal{S}_2 \subseteq \mathcal{A}_2$. The lifting*

$$\mu_1 \ \{(a_1, a_2) \mid a_1 \in \mathcal{S}_1 \to a_2 \in \mathcal{S}_2\}^\sharp \ \mu_2$$

implies $\mu_1(\mathcal{S}_1) \leq \mu_2(\mathcal{S}_2)$. The converse holds when μ_1 and μ_2 have equal weight.

Proof. Let $\mathcal{R}$ be the relation $\{(a_1, a_2) \mid a_1 \in \mathcal{S}_1 \to a_2 \in \mathcal{S}_2\}$. The forward direction is immediate by Theorem 2.1.11, taking the subset $\mathcal{S}_1$. For the reverse direction, consider any non-empty subset $\mathcal{T}_1 \subseteq \mathcal{A}_1$. If $\mathcal{T}_1$ is not contained in $\mathcal{S}_1$, then $\mathcal{R}(\mathcal{T}_1) = \mathcal{A}_2$ and $\mu_1(\mathcal{T}_1) \leq \mu_2(\mathcal{R}(\mathcal{T}_1))$ since μ_1 and μ_2 have equal weight. Otherwise $\mathcal{R}(\mathcal{T}_1) = \mathcal{S}_2$, so

$$\mu_1(\mathcal{T}_1) \leq \mu_1(\mathcal{S}_1) \leq \mu_2(\mathcal{S}_2) = \mu_2(\mathcal{R}(\mathcal{T}_1)).$$

Theorem 2.1.11 gives the desired lifting:

$$\mu_1 \ \{(a_1, a_2) \mid a_1 \in \mathcal{S}_1 \to a_2 \in \mathcal{S}_2\}^\sharp \ \mu_2. \qquad \square$$

A slightly more subtle consequence is *stochastic domination*, an order on distributions over an ordered set.

Definition 2.1.13. *Let $(\mathcal{A}, \leq_{\mathcal{A}})$ be an ordered set and suppose μ_1, μ_2 are sub-distributions over $\mathcal{A}$. We say μ_2 stochastically dominates μ_1, denoted $\mu_1 \leq_{sd} \mu_2$, if*

$$\mu_1(\{a \in \mathcal{A} \mid k \leq_{\mathcal{A}} a\}) \leq \mu_2(\{a \in \mathcal{A} \mid k \leq_{\mathcal{A}} a\})$$

for every $k \in \mathcal{A}$.

This order is different from the pointwise order on sub-distributions since it uses the order on the underlying space. For instance, two proper distributions satisfy $\mu_1 \leq \mu_2$ exactly when $\mu_1 = \mu_2$, but two unequal distributions may satisfy $\mu_1 \leq_{sd} \mu_2$; e.g., if we take distributions over the natural numbers $\mathbb{N}$ with the usual order and μ_1 places weight 1 on 0 while μ_2 places weight 1 on 1.

Stochastic domination is precisely the probabilistic lifting of the order relation.

Proposition 2.1.14. *Suppose μ_1, μ_2 are sub-distributions over a set $\mathcal{A}$ with a reflexive order $\leq_{\mathcal{A}}$ (i.e., $a \leq_{\mathcal{A}} a$). Then $\mu_1 \ (\leq_{\mathcal{A}})^\sharp \ \mu_2$ implies $\mu_1 \leq_{sd} \mu_2$. The converse also holds when μ_1 and μ_2 have equal weight, as long as any upwards closed subset of $\mathcal{A}$ either contains a minimum element or is the whole set $\mathcal{A}$ (e.g., $\mathcal{A} = \mathbb{N}$ or $\mathbb{Z}$ with the usual order).*

Proof. Let $\mathcal{R} \triangleq (\leq_{\mathcal{A}})$. For the forward direction, Theorem 2.1.11 gives

$$\mu_1(\{a \in \mathcal{A} \mid k \leq_{\mathcal{A}} a\}) \leq \mu_2(\mathcal{R}(\{a \in \mathcal{A} \mid k \leq_{\mathcal{A}} a\})).$$

The subset on the right is precisely the set of $a' \in \mathcal{A}$ such that $a' \geq_{\mathcal{A}} a$ for some $a \geq_{\mathcal{A}} k$; by transitivity and reflexivity, we have

$$\mu_2(\mathcal{R}(\{a \in \mathcal{A} \mid k \leq_{\mathcal{A}} a\})) = \mu_2(\{a \in \mathcal{A} \mid k \leq_{\mathcal{A}} a\}).$$

This holds for all $k \in \mathcal{A}$, establishing $\mu_1 \leq_{sd} \mu_2$.

For the converse, suppose $\mu_1 \leq_{sd} \mu_2$ and μ_1 and μ_2 have equal weights, and let $\mathcal{S} \subseteq \mathcal{A}$ be any subset. If the upwards closure $\mathcal{R}(\mathcal{S})$ is the whole set $\mathcal{A}$, then $\mu_1(\mathcal{S}) \leq \mu_2(\mathcal{R}(\mathcal{S}))$ since μ_1 and μ_2 have equal weights. Otherwise, there is a least element k of $\mathcal{R}(\mathcal{S})$ by assumption, and we have

$$\mu_1(\mathcal{S}) \leq \mu_1(\mathcal{R}(\mathcal{S})) = \mu_1(\{a \in \mathcal{A} \mid k \leq_{\mathcal{A}} a\}) \leq \mu_2(\{a \in \mathcal{A} \mid k \leq_{\mathcal{A}} a\}) = \mu_2(\mathcal{R}(\mathcal{S})),$$

where the middle inequality is by stochastic domination. Theorem 2.1.11 implies $\mu_1 \ (\leq_{\mathcal{A}})^\sharp \ \mu_2$. $\qquad \square$

Finally, a typical application of coupling proofs is showing that two distributions are close together.

Definition 2.1.15. Let μ_1, μ_2 be sub-distributions over $\mathcal{A}$. The *total variation distance* (also known as *TV-distance* or *statistical distance*) between μ_1 and μ_2 is defined as

$$d_{\mathrm{tv}}(\mu_1, \mu_2) \triangleq \frac{1}{2} \sum_{a \in \mathcal{A}} |\mu_1(a) - \mu_2(a)| = \max_{\mathcal{S} \subseteq \mathcal{A}} |\mu_1(\mathcal{S}) - \mu_2(\mathcal{S})|.$$

In particular, the total variation distance bounds the difference in probabilities of any event.

Couplings are closely related to TV-distance.

Theorem 2.1.16 (see, e.g., Levin, Peres, and Wilmer (2009); Lindvall (2002)). *Let μ_1 and μ_2 be sub-distributions over $\mathcal{A}$ and let μ be a coupling. Then*

$$d_{tv}(\mu_1, \mu_2) \leq \Pr_{(a_1, a_2) \sim \mu} [a_1 \neq a_2].$$

In particular, if μ witnesses the lifting

$$\mu_1 \; \{(a_1, a_2) \in \mathcal{A} \times \mathcal{A} \mid (a_1, a_2) \in \mathcal{S} \rightarrow a_1 = a_2\}^{\sharp} \; \mu_2,$$

then the TV-distance is bounded by the probability

$$d_{tv}(\mu_1, \mu_2) \leq \Pr_{(a_1, a_2) \sim \mu} [(a_1, a_2) \notin \mathcal{S}].$$

Theorem 2.1.16 is the fundamental result behind the so-called *coupling method* (Aldous, 1983), a technique to show two probabilistic processes converge by constructing a coupling that causes the processes to become equal with high probability.[1] This theorem is usually stated for proper distributions μ_1 and μ_2; the result on sub-distributions follows as an easy consequence. (If there is a lifting then μ_1 and μ_2 have equal weights w by Proposition 2.1.9, and the inequalities in Theorem 2.1.16 are preserved when μ_1 and μ_2 are scaled by the same constant. When $w = 0$ the inequality is immediate; otherwise, by scaling up both distributions by $1/w$, applying the standard theorem to obtain the total variation bound for proper distributions, then scaling back down by w, we recover the total variation bound for sub-distributions.) Unlike the previous facts, the target property about μ_1 and μ_2 does not directly follow from the existence of a lifting—we need more detailed information about the coupling μ.

Proof by coupling

The previous results suggest an indirect approach to proving properties of two distributions: demonstrate there exists a coupling of a particular form. However, how are we supposed to find a witness distribution with the desired properties? The given distributions may be highly complex, possibly over infinite sets—it is not clear how to represent, much less construct, the desired coupling.

To address this challenge, probability theorists have developed a powerful proof technique called *proof by coupling*. This technique assumes a bit more information about the distributions: we need concrete descriptions of two processes *producing* the distributions. Usually, these generating programs are readily available; indeed, they are often the most natural descriptions of complex distributions.

Given two programs, a proof by coupling builds a coupling for the output distributions by coupling intermediate samples. In a bit more detail, we imagine stepping through the programs in parallel, one instruction at a time, starting from two inputs. Whenever we reach two corresponding sampling instructions, we pick a valid coupling for the sampled distributions. The selected couplings induce a relation on samples, which we can assume when analyzing the rest of the programs. For instance, by selecting couplings for earlier samples carefully, we may

---

[1] The converse of Theorem 2.1.16 also holds: there exists a coupling μ_{max}, known as the *maximal* or *optimal coupling*, that achieves equality (see, e.g., Levin et al. (2009); Lindvall (2002)). However, this result will not be important for our purposes.

be able to assume the coupled programs take the same path at a subsequent branching statement; in this way, coupling proofs can consider just pairs of well-behaved executions.

Finding appropriate couplings is the main intellectual challenge when carrying out a proof by coupling, the steps requiring ingenuity. We close this section with an example of the proof technique in action.

Example 2.1.17. Consider a probabilistic process that tosses a fair coin T times and returns the number of heads. If μ_1, μ_2 are the output distributions from running this process for $T = T_1, T_2$ iterations respectively and $T_1 \leq T_2$, then $\mu_1 \leq_{sd} \mu_2$.

Proof by coupling. For the first T_1 iterations, couple the coin flips to be equal—this ensures that after the first T_1 iterations, the coupled counts are equal. The remaining $T_2 - T_1$ coin flips in the second run can only increase the second count, while preserving the first count. Therefore under the coupling, the first count is no more than the second count at termination, establishing $\mu_1 \leq_{sd} \mu_2$.   □

For readers unfamiliar with these proofs, this argument may appear bewildering. The coupling is constructed implicitly, and some of the steps are mysterious. To clarify such proofs, a natural idea is to design a formal logic describing coupling proofs. Somewhat surprisingly, the logic we are looking for was already proposed in the formal verification literature, originally for verifying security of cryptographic protocols.

2.2   A formal logic for coupling proofs

We will work with the logic PRHL (probabilistic Relational Hoare Logic) proposed by Barthe et al. (2009). Before detailing its connection to coupling proofs, we provide a brief introduction to program logics.

Program logics: A brief primer

A *logic* consists of a collection of formulas, also known as *judgments*, and an interpretation describing what it means—in typical, standard mathematics—for judgments to be true (*valid*). While it is possible to prove judgments valid directly by using regular mathematical arguments, this is often inconvenient as the interpretation may be quite complicated. Instead, many logics provide a *proof system*, a set of *logical rules* describing how to combine known judgments (the *premises*) to prove a new judgment (the *conclusion*). Each rule represents a single step in a formal proof. Starting from judgments given by rules with no premises (*axioms*), we can successively apply rules to prove new judgments, building a tree-shaped *derivation* culminating in a single judgment. To ensure that this final judgment is valid, each logical rule should be *sound*: if the premises are valid, then so is the conclusion. Soundness is a basic property, typically one of the first results to be proved about a logic.

Program logics were first introduced by Hoare (1969), building on earlier ideas by Floyd (1967); they are also called *Floyd-Hoare logics*. These logics are really two logics in one: the *assertion logic*, where formulas describe program states, and the program logic proper, where judgments describe imperative programs. A judgment in the main program logic consists of three parts: a program c and two *assertions* Φ and Ψ from the assertion logic. The *pre-condition* Φ describes the initial conditions before executing c (for instance, assumptions about the input), while the *post-condition* Ψ describes the final conditions after executing c (for instance, properties of the output). Hoare (1969) proposed the original logical rules, which construct a judgment for a program by combining judgments for its sub-programs. This compositional style of reasoning is a hallmark of program logics.

By varying the interpretation of judgments, the assertion logic, and the logical rules, Floyd-Hoare logics can establish a variety of properties about different kinds of imperative programs. Notable extensions reason about non-determinism (Dijkstra, 1976), pointers and memory allocation (O'Hearn, Reynolds, and Yang, 2001; Reynolds, 2001, 2002), concurrency (O'Hearn, 2007), and more. (Readers should consult a survey for a more comprehensive account of Floyd-Hoare logic (Apt, 1981, 1983; Jones, 2003).)

In this tradition, Barthe et al. (2009) introduced the logic PRHL targeting security properties in cryptography. Compared to standard program logics, there are two twists: each judgment describes *two* programs, and programs

can use random sampling. In short, PRHL is a *probabilistic Relational Hoare Logic*. Judgments encode probabilistic relational properties of two programs, where a post-condition describes a probabilistic liftings between two output distributions. More importantly, the proof rules represent different ways to combine liftings, formalizing various steps in coupling proofs. Accordingly, we will interpret PRHL as a formal logic for proofs by coupling.

To build up to this connection, we first provide a brief overview of a core version of PRHL, reviewing the programming language, the judgments and their interpretation, and the logical rules.

The logic PRHL: the programming language

Programs in PRHL are defined in terms of *expressions* $\mathcal{E}$ including constants, like the integers and booleans, as well as combinations of constants and variables with primitive operations, like addition and subtraction. We suppose $\mathcal{E}$ also includes terms for basic datatypes, like tuples and lists. Concretely, $\mathcal{E}$ is inductively defined by the following grammar:

$$
\begin{array}{rlll}
\mathcal{E} & ::= & \mathcal{X} \mid \mathcal{L} & \text{(variables)} \\
 & \mid & \mathbb{Z} \mid \mathcal{E} + \mathcal{E} \mid \mathcal{E} - \mathcal{E} \mid \mathcal{E} \cdot \mathcal{E} & \text{(numbers)} \\
 & \mid & \mathbb{B} \mid \mathcal{E} \wedge \mathcal{E} \mid \mathcal{E} \vee \mathcal{E} \mid \neg \mathcal{E} \mid \mathcal{E} = \mathcal{E} \mid \mathcal{E} < \mathcal{E} & \text{(booleans)} \\
 & \mid & (\mathcal{E}, \ldots, \mathcal{E}) \mid \pi_i(\mathcal{E}) \mid [\,] \mid \mathcal{E} :: \mathcal{E} \mid \mathcal{O}(\mathcal{E}) & \text{(tuples, lists, operations)}
\end{array}
$$

Expressions can mention two classes of variables: a countable set $\mathcal{X}$ of *program variables*, which can be modified by the program, and a set $\mathcal{L}$ of *logical variables*, which model fixed parameters. Expressions are typed as numbers, booleans, tuples, or lists, and primitive operations $\mathcal{O}$ have typed signatures; we consider only well-typed expressions throughout. The expressions $(\mathcal{E}, \ldots, \mathcal{E})$ and $\pi_i(\mathcal{E})$ construct and project from a tuple, respectively; $[\,]$ is the empty list, and $\mathcal{E} :: \mathcal{E}$ adds an element to the head of a list. We typically use the letter e for expressions, $x, y, z, \ldots$ for program variables, and lower-case Greek letters $(\alpha, \beta, \ldots)$ and capital Roman letters $(N, M, \ldots)$ for logical variables.

We write $\mathcal{V}$ for the countable set of *values*, including integers, booleans, tuples, finite lists, etc. We can interpret expressions given maps from variables and logical variables to values.

Definition 2.2.1. Program states are *memories*, maps $\mathcal{X} \to \mathcal{V}$; we usually write m for a memory and **State** for the set of memories. *Logical contexts* are maps $\mathcal{L} \to \mathcal{V}$; we usually write ρ for a logical context.

We interpret an expression e as a function $[\![e]\!]_\rho : \textbf{State} \to \mathcal{V}$ in the usual way, for instance:

$$[\![e_1 + e_2]\!]_\rho m \triangleq [\![e_1]\!]_\rho m + [\![e_2]\!]_\rho m.$$

Likewise, we interpret primitive operations o as functions $[\![o]\!]_\rho : \mathcal{V} \to \mathcal{V}$, so that

$$[\![o(e)]\!]_\rho m \triangleq [\![o]\!]_\rho ([\![e]\!]_\rho m).$$

We fix a set $\mathcal{DE}$ of *distribution expressions* to model primitive distributions that our programs can sample from. For simplicity, we suppose for now that each distribution expression d is interpreted as a uniform distribution over a finite set. So, we have the coin flip and uniform distributions:

$$\mathcal{DE} ::= \textbf{Flip} \mid \textbf{Unif}(\mathcal{E})$$

where $\mathcal{E}$ is a list, representing the space of samples. We will introduce other primitive distributions as needed. To interpret distribution expressions, we define $[\![d]\!]_\rho : \textbf{State} \to \textbf{Distr}(\mathcal{V})$; for instance,

$$[\![\textbf{Unif}(e)]\!]_\rho m \triangleq \mathcal{U}([\![e]\!]_\rho m)$$

where $\mathcal{U}(\mathcal{S})$ is the mathematical uniform distribution over a set $\mathcal{S}$.

Now let's see the programming language. We work with a standard imperative language with random sampling. The programs, also called *commands* or *statements*, are defined inductively:

$$
\begin{array}{rcll}
\mathcal{C} & := & \mathbf{skip} & \text{(no-op)} \\
& | & \mathcal{X} \leftarrow \mathcal{E} & \text{(assignment)} \\
& | & \mathcal{X} \xleftarrow{\$} \mathcal{D}\mathcal{E} & \text{(sampling)} \\
& | & \mathcal{C}; \mathcal{C} & \text{(sequencing)} \\
& | & \mathbf{if}\ \mathcal{E}\ \mathbf{then}\ \mathcal{C}\ \mathbf{else}\ \mathcal{C} & \text{(conditional)} \\
& | & \mathbf{while}\ \mathcal{E}\ \mathbf{do}\ \mathcal{C} & \text{(loop)}
\end{array}
$$

We assume throughout that programs are well-typed; for instance, the guard expressions in conditionals and loops must be boolean.

We interpret each command as a mathematical function from states to sub-distributions over output states; this function is known as the *semantics* of a command. Since the set of program variables and the set of values are countable, the set of states is also countable so sub-distributions over states are discrete. To interpret commands, we use two basic constructions on sub-distributions.

Definition 2.2.2. The function $\mathbf{unit} : \mathcal{A} \to \mathbf{SDistr}(\mathcal{A})$ maps every element $a \in \mathcal{A}$ to the sub-distribution that places probability 1 on a. The function $\mathbf{bind} : \mathbf{SDistr}(\mathcal{A}) \times (\mathcal{A} \to \mathbf{SDistr}(\mathcal{B})) \to \mathbf{SDistr}(\mathcal{B})$ is defined by

$$
\mathbf{bind}(\mu, f)(b) \triangleq \sum_{a \in \mathcal{A}} \mu(a) \cdot f(a)(b).
$$

Intuitively, **bind** applies a randomized function on a distribution over inputs.

We use a discrete version of the semantics considered by Kozen (1981), presented in Fig. 2.1; we write $m[x \mapsto v]$ for the memory m with variable x updated to hold v, and $a \mapsto b(a)$ for the function mapping a to $b(a)$. The most complicated case is for loops. The sub-distribution $\mu^{(i)}(m)$ models executions that exit after entering the loop body at most i times, starting from initial memory m. For the base case $i = 0$, the sub-distribution either returns m with probability 1 when the guard is false and the loop exits immediately, or returns the null sub-distribution $\perp$ when the guard is true. The cases $i > 0$ are defined recursively, by unrolling the loop.

Note that $\mu^{(i)}$ are increasing in i: $\mu^{(i)}(m) \leq \mu^{(j)}(m)$ for all $m \in \mathbf{State}$ and $i \leq j$. In particular, the weights of the sub-distributions are increasing. Since the weights are at most 1, the approximants converge to a sub-distribution as i tends to infinity by the monotone convergence theorem (see, e.g., Rudin (1976, Theorem 11.28), taking the discrete (counting) measure over **State**).

The logic PRHL: judgments and validity

The program logic PRHL features judgments of the following form:

$$
c_1 \sim c_2 : \Phi \Longrightarrow \Psi
$$

Here, c_1 and c_2 are commands and Φ and Ψ are predicates on pairs of memories. To describe the inputs and outputs of c_1 and c_2, each predicate can mention two copies $x\langle 1 \rangle$, $x\langle 2 \rangle$ of each program variable x; these *tagged* variables refer to the value of x in the executions of c_1 and c_2 respectively.

Definition 2.2.3. Let $\mathcal{X}\langle 1 \rangle$ and $\mathcal{X}\langle 2 \rangle$ be the sets of *tagged variables*, finite sets of variable names tagged with $\langle 1 \rangle$ or $\langle 2 \rangle$ respectively:

$$
\mathcal{X}\langle 1 \rangle \triangleq \{x\langle 1 \rangle \mid x \in \mathcal{X}\} \quad \text{and} \quad \mathcal{X}\langle 2 \rangle \triangleq \{x\langle 2 \rangle \mid x \in \mathcal{X}\}.
$$

$$\llbracket \textbf{skip} \rrbracket_\rho\, m \triangleq \textbf{unit}(m)$$

$$\llbracket x \leftarrow e \rrbracket_\rho\, m \triangleq \textbf{unit}(m[x \mapsto \llbracket e \rrbracket_\rho m])$$

$$\llbracket x \xleftarrow{\$} d \rrbracket_\rho\, m \triangleq \textbf{bind}(\llbracket d \rrbracket_\rho m, v \mapsto \textbf{unit}(m[x \mapsto v]))$$

$$\llbracket c;\, c' \rrbracket_\rho\, m \triangleq \textbf{bind}(\llbracket c \rrbracket_\rho m, \llbracket c' \rrbracket_\rho)$$

$$\llbracket \textbf{if } e \textbf{ then } c \textbf{ else } c' \rrbracket_\rho\, m \triangleq \begin{cases} \llbracket c \rrbracket_\rho m & : \llbracket e \rrbracket_\rho m = \textbf{true} \\ \llbracket c' \rrbracket_\rho m & : \llbracket e \rrbracket_\rho m = \textbf{false} \end{cases}$$

$$\llbracket \textbf{while } e \textbf{ do } c \rrbracket_\rho\, m \triangleq \lim_{i \to \infty} \mu^{(i)}(m)$$

$$\mu^{(i)}(m) \triangleq \begin{cases} \bot & : i = 0 \wedge \llbracket e \rrbracket_\rho m = \textbf{true} \\ \textbf{unit}(m) & : i = 0 \wedge \llbracket e \rrbracket_\rho m = \textbf{false} \\ \textbf{bind}(\llbracket \textbf{if } e \textbf{ then } c \rrbracket_\rho m, \mu^{(i-1)}) & : i > 0 \end{cases}$$

Figure 2.1: Semantics of programs

Let $\textbf{State}\langle 1 \rangle$ and $\textbf{State}\langle 2 \rangle$ be the sets of *tagged memories*, maps from tagged variables to values:

$$\textbf{State}\langle 1 \rangle \triangleq \mathcal{X}\langle 1 \rangle \to \mathcal{V} \quad \text{and} \quad \textbf{State}\langle 2 \rangle \triangleq \mathcal{X}\langle 2 \rangle \to \mathcal{V}.$$

Let $\textbf{State}_\times$ be the set of *product memories*, which combine two tagged memories:

$$\textbf{State}_\times \triangleq \mathcal{X}\langle 1 \rangle \uplus \mathcal{X}\langle 2 \rangle \to \mathcal{V}.$$

For notational convenience we identify $\textbf{State}_\times$ with pairs of memories $\textbf{State}\langle 1 \rangle \times \textbf{State}\langle 2 \rangle$; for $m_1 \in \textbf{State}\langle 1 \rangle$ and $m_2 \in \textbf{State}\langle 2 \rangle$, we write (m_1, m_2) for the product memory and we use the usual projections on pairs to extract untagged memories from the product memory:

$$p_1(m_1, m_2) \triangleq |m_1| \quad \text{and} \quad p_2(m_1, m_2) \triangleq |m_2|,$$

where the memory $|m| \in \textbf{State}$ has all variables in $\mathcal{X}$. For commands c and expressions e with variables in $\mathcal{X}$, we write $c\langle 1 \rangle$, $c\langle 2 \rangle$ and $e\langle 1 \rangle$, $e\langle 2 \rangle$ for the corresponding *tagged commands* and *tagged expressions* with variables in $\mathcal{X}\langle 1 \rangle$ and $\mathcal{X}\langle 2 \rangle$.

We consider a set $\mathcal{P}$ of *predicates* (*assertions*) from first-order logic defined by the following grammar:

$$
\begin{array}{rcll}
\mathcal{P} & ::= & \mathcal{E}\langle 1/2 \rangle = \mathcal{E}\langle 1/2 \rangle \;\mid\; \mathcal{E}\langle 1/2 \rangle < \mathcal{E}\langle 1/2 \rangle \;\mid\; \mathcal{E}\langle 1/2 \rangle \in \mathcal{E}\langle 1/2 \rangle & \\
& \mid & \top \;\mid\; \bot \;\mid\; \mathcal{O}(\mathcal{E}\langle 1/2 \rangle, \dots, \mathcal{E}\langle 1/2 \rangle) & \text{(predicates)} \\
& \mid & \mathcal{P} \wedge \mathcal{P} \;\mid\; \mathcal{P} \vee \mathcal{P} \;\mid\; \neg \mathcal{P} \;\mid\; \mathcal{P} \to \mathcal{P} \;\mid\; \forall \mathcal{L} \in \mathbb{Z},\, \mathcal{P} \;\mid\; \exists \mathcal{L} \in \mathbb{Z},\, \mathcal{P} & \text{(first-order formulas)}
\end{array}
$$

We typically use capital Greek letters ($\Phi, \Psi, \Theta, \Xi, \dots$) for predicates. $\mathcal{E}\langle 1/2 \rangle$ denotes an expression where program variables are tagged with $\langle 1 \rangle$ or $\langle 2 \rangle$; tags may be mixed within an expression. We consider the usual binary predicates $\{=, <, \in, \dots\}$ where $e \in e'$ means e is a member of the list e', and we take the always-true and always-false predicates $\top$ and $\bot$, and a set $\mathcal{O}$ of other predicates. Predicates can be combined using the usual connectives $\{\wedge, \vee, \neg, \to\}$ and can quantify over first-order types (e.g., the integers, tuples, etc.). We will often interpret a boolean expression e as the predicate $e = \textbf{true}$.

Predicates are interpreted as sets of product memories.

Definition 2.2.4. Let Φ be a predicate. Given a logical context ρ, Φ is interpreted as a set $[\![\Phi]\!]_\rho \subseteq \mathbf{State}_\times$ in the expected way, e.g.,

$$[\![e_1\langle 1\rangle < e_2\langle 2\rangle]\!]_\rho \triangleq \{(m_1, m_2) \in \mathbf{State}_\times \mid [\![e_1]\!]_\rho m_1 < [\![e_2]\!]_\rho m_2\}.$$

We can inject a predicate on single memories into a predicate on product memories; we call the resulting predicate *one-sided* since it constrains just one of two memories.

Definition 2.2.5. Let Φ be a predicate on **State**. We define formulas $\Phi\langle 1\rangle$ and $\Phi\langle 2\rangle$ by replacing all program variables x in Φ with $x\langle 1\rangle$ and $x\langle 2\rangle$, respectively, and we define

$$[\![\Phi\langle 1\rangle]\!]_\rho \triangleq \{(m_1, m_2) \mid m_1 \in [\![\Phi]\!]_\rho\} \quad \text{and} \quad [\![\Phi\langle 2\rangle]\!]_\rho \triangleq \{(m_1, m_2) \mid m_2 \in [\![\Phi]\!]_\rho\}.$$

Valid judgments in PRHL relate two output distributions by lifting the post-condition.

Definition 2.2.6 (Barthe et al. (2009))**.** A judgment is *valid* in logical context ρ, written $\rho \models c_1 \sim c_2 : \Phi \Longrightarrow \Psi$, if for any two memories $(m_1, m_2) \in [\![\Phi]\!]_\rho$ there exists a lifting of Ψ relating the output distributions:

$$[\![c_1]\!]_\rho m_1 \; [\![\Psi]\!]_\rho^\sharp \; [\![c_2]\!]_\rho m_2.$$

For example, a valid judgment

$$\models c_1 \sim c_2 : \Phi \Longrightarrow (=),$$

states that for any two input memories (m_1, m_2) satisfying Φ, the resulting output distributions from running c_1 and c_2 are related by lifted equality; by Proposition 2.1.10, these output distributions must be equal.

The logic PRHL: the proof rules

The logic PRHL includes a collection of logical rules to inductively build up a proof of a new judgment from known judgments. The rules are superficially similar to those from standard Hoare logic. However, the interpretation of judgments in terms of liftings means some rules in PRHL are not valid in Hoare logic, and vice versa.

Before describing the rules, we introduce some necessary notation. A system of logical rules inductively defines a set of *derivable* formulas; we use the head symbol $\vdash$ to mark such formulas. The premises in each logical rule are written above the horizontal line, and the single conclusion is written below the line; for easy reference, the name of each rule is given to the left of the line.

The main premises are judgments in the program logic, but rules may also use other *side-conditions*. For instance, many rules require an assertion logic formula to be valid in all memories. Other side-conditions state that a program is terminating, or that certain variables are not modified by the program. We use the head symbol $\models$ to mark valid side-conditions; while we could give a separate proof system for these premises, in practice they are simple enough to check directly.

We also use notation for substitution in assertions. We write $\Phi\{e/x\}$ for the formula Φ with every occurrence of the variable x replaced by e. Similarly, $\Phi\{v_1, v_2/x_1\langle 1\rangle, x_2\langle 2\rangle\}$ is the formula Φ where occurrences of the tagged variables $x_1\langle 1\rangle, x_2\langle 2\rangle$ are replaced by v_1, v_2 respectively.

The rules of PRHL can be divided into three groups: *two-sided* rules, *one-sided* rules, and *structural* rules. All judgments are parameterized by a logical context ρ, but since this context is assumed to be a fixed assignment of logical variables—constant throughout the proof—we omit it from the rules. The two-sided rules in Fig. 2.2 apply when the two programs in the conclusion judgment have the same top-level shape.

The rule [SKIP] simply states that **skip** instructions preserve the pre-condition. The rule [ASSN] handles assignment instructions. It is the usual Hoare-style rule: if Ψ holds initially with $e_1\langle 1\rangle$ and $e_2\langle 2\rangle$ substituted for $x_1\langle 1\rangle$ and $x_2\langle 2\rangle$, then Ψ holds after the respective assignment instructions.

$$\text{SKIP} \; \frac{}{\vdash \mathbf{skip} \sim \mathbf{skip} : \Phi \Longrightarrow \Phi}$$

$$\text{ASSN} \; \frac{}{\vdash x_1 \leftarrow e_1 \sim x_2 \leftarrow e_2 : \Psi \{e_1\langle 1\rangle, e_2\langle 2\rangle / x_1\langle 1\rangle, x_2\langle 2\rangle\} \Longrightarrow \Psi}$$

$$\text{SAMPLE} \; \frac{f : \mathrm{supp}(d_1) \rightarrow \mathrm{supp}(d_2) \text{ is a bijection}}{\vdash x_1 \xleftarrow{\$} d_1 \sim x_2 \xleftarrow{\$} d_2 : \forall v \in \mathrm{supp}(d_1), \; \Psi \{v, f(v)/x_1\langle 1\rangle, x_2\langle 2\rangle\} \Longrightarrow \Psi}$$

$$\text{SEQ} \; \frac{\vdash c_1 \sim c_2 : \Phi \Longrightarrow \Psi \qquad \vdash c_1' \sim c_2' : \Psi \Longrightarrow \Theta}{\vdash c_1; c_1' \sim c_2; c_2' : \Phi \Longrightarrow \Theta}$$

$$\text{COND} \; \frac{\models \Phi \rightarrow e_1\langle 1\rangle = e_2\langle 2\rangle \qquad \vdash c_1 \sim c_2 : \Phi \wedge e_1\langle 1\rangle \Longrightarrow \Psi \qquad \vdash c_1' \sim c_2' : \Phi \wedge \neg e_1\langle 1\rangle \Longrightarrow \Psi}{\vdash \mathbf{if} \; e_1 \; \mathbf{then} \; c_1 \; \mathbf{else} \; c_1' \sim \mathbf{if} \; e_2 \; \mathbf{then} \; c_2 \; \mathbf{else} \; c_2' : \Phi \Longrightarrow \Psi}$$

$$\text{WHILE} \; \frac{\models \Phi \rightarrow e_1\langle 1\rangle = e_2\langle 2\rangle \qquad \vdash c_1 \sim c_2 : \Phi \wedge e_1\langle 1\rangle \Longrightarrow \Phi}{\vdash \mathbf{while} \; e_1 \; \mathbf{do} \; c_1 \sim \mathbf{while} \; e_2 \; \mathbf{do} \; c_2 : \Phi \Longrightarrow \Phi \wedge \neg e_1\langle 1\rangle}$$

Figure 2.2: Two-sided PRHL rules

The rule [SAMPLE] is more subtle. In some ways it is the key rule in PRHL, allowing us to select a coupling for a pair of sampling instructions. To gain intuition, the following rule is a special case:

$$\text{SAMPLE*} \; \frac{f : \mathrm{supp}(d) \rightarrow \mathrm{supp}(d) \text{ is a bijection}}{\vdash x \xleftarrow{\$} d \sim x \xleftarrow{\$} d : \top \Longrightarrow f(x\langle 1\rangle) = x\langle 2\rangle}$$

The conclusion states that there exists a coupling of a distribution d with itself such that each sample x from d is related to $f(x)$. Soundness of this rule crucially relies on d being *uniform*—as we have seen, any bijection f induces a coupling of uniform distributions (cf. Example 2.1.4). It is possible to support general distributions at the cost of a more complicated side-condition,[2] but we will not need this generality. The full rule [SAMPLE] can prove a post-condition of any shape: a post-condition holds after sampling if it holds before sampling, where $x\langle 1\rangle$ and $x\langle 2\rangle$ are replaced by any two coupled samples $(v, f(v))$.

The rule [SEQ] resembles the normal rule for sequential composition in Hoare logic, but its reading is more subtle. In particular, note that the intermediate assertion Ψ is interpreted differently in the two premises: in the first judgment it is a post-condition and interpreted as a relation between *distributions over memories* via lifting, while in the second judgment it is a pre-condition and interpreted as a relation between *memories*.

The next two rules deal with branching commands. Rule [COND] requires that the guards $e_1\langle 1\rangle$ and $e_2\langle 2\rangle$ are equal assuming the pre-condition Φ. The rule is otherwise similar to the standard Hoare logic rule: if we can prove the post-condition Ψ when the guard is initially true and when the guard is initially false, then we can prove Ψ as a post-condition of the conditional.

Rule [WHILE] uses a similar idea for loops. We again assume that the guards are initially equal, and we also assume that they are equal in the post-condition of the loop body. Since the judgments are interpreted in terms of couplings, this second condition is a bit subtle. For one thing, the rule does *not* require $e_1\langle 1\rangle = e_2\langle 2\rangle$ in all possible

---

[2]Roughly speaking, the probability of any set S under d should be equal to the probability of $f(S)$ under d.

$$\text{Assn-L} \ \frac{}{\vdash x_1 \leftarrow e_1 \sim \mathbf{skip} : \Psi \{e_1\langle 1\rangle / x_1\langle 1\rangle\} \Longrightarrow \Psi}$$

$$\text{Assn-R} \ \frac{}{\vdash \mathbf{skip} \sim x_2 \leftarrow e_2 : \Psi \{e_2\langle 2\rangle / x_2\langle 2\rangle\} \Longrightarrow \Psi}$$

$$\text{Sample-L} \ \frac{}{\vdash x_1 \xleftarrow{\$} d_1 \sim \mathbf{skip} : \forall v \in \mathrm{supp}(d_1),\ \Psi \{v / x_1\langle 1\rangle\} \Longrightarrow \Psi}$$

$$\text{Sample-R} \ \frac{}{\vdash \mathbf{skip} \sim x_2 \xleftarrow{\$} d_2 : \forall v \in \mathrm{supp}(d_2),\ \Psi \{v / x_2\langle 2\rangle\} \Longrightarrow \Psi}$$

$$\text{Cond-L} \ \frac{\vdash c_1 \sim c : \Phi \wedge e_1\langle 1\rangle \Longrightarrow \Psi \qquad \vdash c_1' \sim c : \Phi \wedge \neg e_1\langle 1\rangle \Longrightarrow \Psi}{\vdash \mathbf{if}\ e_1\ \mathbf{then}\ c_1\ \mathbf{else}\ c_1' \sim c : \Phi \Longrightarrow \Psi}$$

$$\text{Cond-R} \ \frac{\vdash c \sim c_2 : \Phi \wedge e_2\langle 2\rangle \Longrightarrow \Psi \qquad \vdash c \sim c_2' : \Phi \wedge \neg e_2\langle 2\rangle \Longrightarrow \Psi}{\vdash c \sim \mathbf{if}\ e_2\ \mathbf{then}\ c_2\ \mathbf{else}\ c_2' : \Phi \Longrightarrow \Psi}$$

$$\text{While-L} \ \frac{\vdash c_1 \sim \mathbf{skip} : \Phi \wedge e_1\langle 1\rangle \Longrightarrow \Phi \qquad \models \Phi \rightarrow \Phi_1\langle 1\rangle \qquad \Phi_1 \models \mathbf{while}\ e_1\ \mathbf{do}\ c_1\ \text{lossless}}{\vdash \mathbf{while}\ e_1\ \mathbf{do}\ c_1 \sim \mathbf{skip} : \Phi \Longrightarrow \Phi \wedge \neg e_1\langle 1\rangle}$$

$$\text{While-R} \ \frac{\vdash \mathbf{skip} \sim c_2 : \Phi \wedge e_2\langle 2\rangle \Longrightarrow \Phi \qquad \models \Phi \rightarrow \Phi_2\langle 2\rangle \qquad \Phi_2 \models \mathbf{while}\ e_2\ \mathbf{do}\ c_2\ \text{lossless}}{\vdash \mathbf{skip} \sim \mathbf{while}\ e_2\ \mathbf{do}\ c_2 : \Phi \Longrightarrow \Phi \wedge \neg e_2\langle 2\rangle}$$

Figure 2.3: One-sided PRHL rules

executions of the two programs—this would be a rather severe restriction, for instance ruling out programs where $e_1\langle 1\rangle$ and $e_2\langle 2\rangle$ are probabilistic. Rather, the guards only need to be equal *under the coupling of the two programs given by the premise*. The upshot is that by selecting appropriate couplings in the loop body, we can assume the guards are equal when analyzing loops with probabilistic guards. The rule is otherwise similar to the usual Hoare logic rule, where Φ is the loop invariant.

So far, we have seen rules that relate two programs of the same shape. These are the most commonly used rules in PRHL, as relational reasoning is most powerful when comparing two highly similar (or even the same) programs. However, in some cases we may need to reason about two programs with different shapes, even if the two top-level commands are the same. For instance, if we can't guarantee two executions of a program follow the same path at a conditional statement under a coupling, we must relate the two different branches. For this kind of reasoning, we can fall back on the *one-sided* rules in Fig. 2.3. These rules relate a command of a particular shape with **skip** or an arbitrary command. Each rule comes in a left- and a right-side version.

The assignment rules, [Assn-L] and [Assn-R], relate an assignment instruction to **skip** using the usual Hoare rule for assignment instructions. The sampling rules, [Sample-L] and [Sample-R], are similar; they relate a

$$\text{CONSEQ} \; \frac{\vdash c_1 \sim c_2 : \Phi' \Longrightarrow \Psi' \qquad \models \Phi \to \Phi' \qquad \models \Psi' \to \Psi}{\vdash c_1 \sim c_2 : \Phi \Longrightarrow \Psi}$$

$$\text{EQUIV} \; \frac{\vdash c_1' \sim c_2' : \Phi \Longrightarrow \Psi \qquad c_1 \equiv c_1' \qquad c_2 \equiv c_2'}{\vdash c_1 \sim c_2 : \Phi \Longrightarrow \Psi}$$

$$\text{CASE} \; \frac{\vdash c_1 \sim c_2 : \Phi \wedge \Theta \Longrightarrow \Psi \qquad \vdash c_1 \sim c_2 : \Phi \wedge \neg \Theta \Longrightarrow \Psi}{\vdash c_1 \sim c_2 : \Phi \Longrightarrow \Psi}$$

$$\text{TRANS} \; \frac{\vdash c_1 \sim c_2 : \Phi \Longrightarrow \Psi \qquad \vdash c_2 \sim c_3 : \Phi' \Longrightarrow \Psi'}{\vdash c_1 \sim c_3 : \Phi' \circ \Phi \Longrightarrow \Psi' \circ \Psi}$$

$$\text{FRAME} \; \frac{\vdash c_1 \sim c_2 : \Phi \Longrightarrow \Psi \qquad \text{FV}(\Theta) \cap \text{MV}(c_1, c_2) = \varnothing}{\vdash c_1 \sim c_2 : \Phi \wedge \Theta \Longrightarrow \Psi \wedge \Theta}$$

Figure 2.4: Structural PRHL rules

sampling instruction to **skip** if the post-condition holds for all possible values of the sample. These rules represent couplings where fresh randomness is used, i.e., where randomness is not shared between the two programs.

The conditional rules, [COND-L] and [COND-R], are similar to the two-sided conditional rule except there is no assumption of synchronized guards—the other command c might not even be a conditional. If we can relate the general command c to the true branch when the guard is true and relate c to the false branch when the guard is false, then we can relate c to the whole conditional.

The rules for loops, [WHILE-L] and [WHILE-R], can only relate loops to the **skip**; a loop that executes multiple iterations cannot be directly related to an arbitrary command that executes only once. These rules mimic the usual loop rule from Hoare logic, with a critical side-condition: *losslessness*.

Definition 2.2.7. A command c is Φ-*lossless* if for any memory m satisfying Φ and every logical context ρ, the output $[\![c]\!]_\rho m$ is a proper distribution (i.e., it has total probability 1). We write Φ-lossless as the following judgment:

$$\Phi \models c \text{ lossless}$$

Losslessness is needed for soundness: **skip** produces a proper distribution on any input and liftings can only relate sub-distributions with equal weights (Proposition 2.1.9), so the loop must also produce a proper distribution to have any hope of coupling the output distributions. For the examples we will consider, losslessness is easy to show since loops execute for a finite number of iterations; when there is no finite bound, proving losslessness may require more sophisticated techniques (e.g., Barthe, Espitau, Gaboardi, Grégoire, Hsu, and Strub (2017a); Chatterjee, Fu, and Goharshady (2016a); Chatterjee, Fu, Novotný, and Hasheminezhad (2016b); Chatterjee, Novotný, and Žikelić (2017); Ferrer Fioriti and Hermanns (2015); McIver, Morgan, Kaminski, and Katoen (2018)).

Finally, PRHL includes a handful of *structural* rules which apply to programs of any shape. The first rule [CONSEQ] is the usual rule of consequence, allowing us to strengthen the pre-condition and weaken the post-condition—assuming more about the input and proving less about the output, respectively.

The rule [EQUIV] replaces programs by equivalent programs. This rule is particularly useful for reasoning about programs of different shapes. Instead of using one-sided rules, which are often less convenient, we can sometimes replace a program with an equivalent version and then apply two-sided rules. For simplicity, we use a

strong notion of equivalence:

$$c_1 \equiv c_2 \triangleq [\![c_1]\!]_\rho = [\![c_2]\!]_\rho$$

for every logical context ρ; more refined notions of equivalence are also possible, but will not be needed for our purposes. For our examples, we just use a handful of basic program equivalences, e.g., $c; \mathbf{skip} \equiv c$ and $\mathbf{skip}; c \equiv c$.

The rule [CASE] performs a case analysis on the input. If we can prove a judgment when Θ holds initially and a judgment when Θ does not hold initially, then we can combine the two judgments provided they have the same post-condition.

The rule [TRANS] is the transitivity rule: given a judgment relating $c_1 \sim c_2$ and a judgment relating $c_2 \sim c_3$, we can glue these judgments together to relate $c_1 \sim c_3$. The pre- and post-conditions of the conclusion are given by composing the pre- and post-conditions of the premises; for binary relations $\mathcal{R}$ and $\mathcal{S}$, relation composition is defined by

$$\mathcal{R} \circ \mathcal{S} \triangleq \{(x_1, x_3) \mid \exists x_2. \, (x_1, x_2) \in \mathcal{S} \wedge (x_2, x_3) \in \mathcal{R}\}.$$

The last rule [FRAME] is the frame rule (also called the *rule of constancy*): it states that an assertion Θ can be carried from the pre-condition through to the post-condition as long as the variables $\mathrm{MV}(c_1, c_2)$ that may be modified by the programs c_1 and c_2 don't include any of the variables $\mathrm{FV}(\Theta)$ appearing free in Θ; as usual, MV and FV are defined syntactically by collecting the variables that occur in programs and assertions.

As expected, the proof system of PRHL is sound.

Theorem 2.2.8 (Barthe et al. (2009)). *Let ρ be a logical context. If a judgment is derivable*

$$\rho \vdash c_1 \sim c_2 : \Phi \Longrightarrow \Psi,$$

then it is valid:

$$\rho \models c_1 \sim c_2 : \Phi \Longrightarrow \Psi.$$

The coupling interpretation

A valid judgment $\rho \models c_1 \sim c_2 : \Phi \Longrightarrow \Psi$ implies that for any two input memories related by Φ, there exists a coupling with support in Ψ between the two output distributions. By applying the results in Section 2.1, valid judgments imply relational properties of programs.

Moreover, by viewing the rules as the discrete steps in a proof, we can identify common pieces of standard coupling proofs. For instance, [SAMPLE] selects a coupling for corresponding sampling statements; the function f lets us choose among different bijection couplings. The rule [SEQ] encodes a composition principle for couplings; when two processes produce samples related by Ψ under a particular coupling, we can continue to assume this relation when analyzing the remainder of the program. The structural rule [CASE] shows we can select between two possible couplings depending on whether a predicate Θ holds. In short, not only is PRHL a logic for verifying cryptographic protocols, it is also a formal logic for proofs by coupling.

2.3 Constructing couplings, formally

Now let's see how to construct coupling proofs in the logic. We give three examples proving classical probabilistic properties: equivalence, stochastic domination, and convergence.

Remark 2.3.1. There are some inherent challenges in presenting formal proofs on paper. Fundamentally, our proofs are branching derivation trees. When such a proof is serialized, it may be hard to follow which part of the derivation tree the paper proof corresponds to. To help organize the proof, we proceed loosely in a top-down fashion, giving proofs and judgments for the most deeply nested parts of the program first and then gradually zooming out to consider larger and larger parts of the whole program.

Applications of sequential composition are also natural places to signpost the proof; we typically consider the commands in order, unless the second command is much more complex than the first. Finally, for space reasons we will gloss over applications of the assignment rule [ASSN] and minor uses of the rule of consequence [CONSEQ]; a completely formal proof would also spell out these details.

Probabilistic equivalence

To warm up, we prove two programs probabilistically equivalent. Our example models perhaps the most basic encryption scheme: the XOR cipher. Given a boolean s representing the secret message, the XOR cipher flips a fair coin to draw the secret key k and then returns $k \oplus s$ as the encrypted message. A receiving party who knows the secret key can decrypt the message by computing $k \oplus (k \oplus s) = s$.

To prove secrecy of this scheme, we consider the following two programs:

$$k \xleftarrow{\$} \textbf{Flip}; \qquad\qquad\qquad k \xleftarrow{\$} \textbf{Flip};$$
$$r \leftarrow k \oplus s \qquad\qquad\qquad\qquad r \leftarrow k$$

The first program xor_1 implements the encryption function, storing the encrypted message into r. The second program xor_2 simply stores a random value into r. If we can show the distribution of r is the same in both programs, then the XOR cipher is secure: the distribution on outputs is completely random, leaking no information about the secret message s. In terms of PRHL, it suffices to prove the following judgment:

$$\vdash xor_1 \sim xor_2 : \top \Longrightarrow r\langle 1 \rangle = r\langle 2 \rangle$$

By validity of the logic, this judgment implies that for any two memories m_1, m_2, the output distributions are related by a coupling that always returns outputs with equal values of r; by reasoning similar to Proposition 2.1.10, this implies that the output distributions over $r\langle 1 \rangle$ and $r\langle 2 \rangle$ are equal.[3]

Before proving this judgment in the logic, we sketch the proof by coupling. If $s\langle 1 \rangle$ is true, then we couple k to take opposite values in the two runs. If $s\langle 1 \rangle$ is false, then we couple k to be equal in the two runs. In both cases, we conclude that the results $r\langle 1 \rangle, r\langle 2 \rangle$ are equal under the coupling.

To formalize this argument in PRHL, we use the [CASE] rule:

$$\text{CASE} \dfrac{\vdash xor_1 \sim xor_2 : s\langle 1 \rangle = \textbf{true} \Longrightarrow r\langle 1 \rangle = r\langle 2 \rangle \qquad \vdash xor_1 \sim xor_2 : s\langle 1 \rangle \neq \textbf{true} \Longrightarrow r\langle 1 \rangle = r\langle 2 \rangle}{\vdash xor_1 \sim xor_2 : \top \Longrightarrow r\langle 1 \rangle = r\langle 2 \rangle} \; .$$

For the first premise we select the negation coupling using the bijection $f = \neg$ in [SAMPLE], apply the assignment rule [ASSN], and combine with the sequencing rule [SEQ]. Concretely, we have

$$\text{SAMPLE} \dfrac{f = \neg}{\vdash k \xleftarrow{\$} \textbf{Flip} \sim k \xleftarrow{\$} \textbf{Flip} : s\langle 1 \rangle = \textbf{true} \Longrightarrow k\langle 1 \rangle = \neg k\langle 2 \rangle \land s\langle 1 \rangle = \textbf{true}}$$

$$\text{ASSN} \dfrac{}{\vdash r \leftarrow k \oplus s \sim r \leftarrow k : k\langle 1 \rangle = \neg k\langle 2 \rangle \land s\langle 1 \rangle = \textbf{true} \Longrightarrow r\langle 1 \rangle = r\langle 2 \rangle}$$

and we combine the two judgments to give:

$$\text{SEQ} \dfrac{\vdash k \xleftarrow{\$} \textbf{Flip} \sim k \xleftarrow{\$} \textbf{Flip} : s\langle 1 \rangle = \textbf{true} \Longrightarrow k\langle 1 \rangle = \neg k\langle 2 \rangle \land s\langle 1 \rangle = \textbf{true} \qquad \vdash r \leftarrow k \oplus s \sim r \leftarrow k : k\langle 1 \rangle = \neg k\langle 2 \rangle \land s\langle 1 \rangle = \textbf{true} \Longrightarrow r\langle 1 \rangle = r\langle 2 \rangle}{\vdash xor_1 \sim xor_2 : s\langle 1 \rangle = \textbf{true} \Longrightarrow r\langle 1 \rangle = r\langle 2 \rangle} \; .$$

---

[3]To be completely precise, Proposition 2.1.10 assumes that we have lifted equality, while here we only have a lifting where the variables r are equal. An analogous argument shows that the marginal distributions of variable r must be equal.

For the other case $s\langle 1\rangle \neq$ **true**, we give the same proof except with the identity coupling in $[\text{SAMPLE}]$:

$$\text{SAMPLE} \frac{f = \text{id}}{\vdash k \xleftarrow{\$} \textbf{Flip} \sim k \xleftarrow{\$} \textbf{Flip} : s\langle 1\rangle \neq \textbf{true} \Longrightarrow k\langle 1\rangle = k\langle 2\rangle \wedge s\langle 1\rangle \neq \textbf{true}}$$

and the assignment rule, we have

$$\text{ASSN} \frac{}{\vdash r \leftarrow k \oplus s \sim r \leftarrow k : k\langle 1\rangle = k\langle 2\rangle \wedge s\langle 1\rangle \neq \textbf{true} \Longrightarrow r\langle 1\rangle = r\langle 2\rangle .}$$

Combining the conclusions, we get

$$\text{SEQ} \frac{\vdash k \xleftarrow{\$} \textbf{Flip} \sim k \xleftarrow{\$} \textbf{Flip} : s\langle 1\rangle \neq \textbf{true} \Longrightarrow k\langle 1\rangle = \neg k\langle 2\rangle \wedge s\langle 1\rangle \neq \textbf{true} \qquad \vdash r \leftarrow k \oplus s \sim r \leftarrow k : k\langle 1\rangle = k\langle 2\rangle \wedge s\langle 1\rangle \neq \textbf{true} \Longrightarrow r\langle 1\rangle = r\langle 2\rangle}{\vdash xor_1 \sim xor_2 : s\langle 1\rangle \neq \textbf{true} \Longrightarrow r\langle 1\rangle = r\langle 2\rangle} .$$

By $[\text{CASE}]$, we conclude the desired post-condition $r\langle 1\rangle = r\langle 2\rangle$.

Stochastic domination

For our second example, we revisit Example 2.1.17 and replicate the proof in PRHL. The following program *sdom* flips a coin T times and returns the number of coin flips that come up true:

$$i \leftarrow 0; ct \leftarrow 0;$$
$$\textbf{while } i < T \textbf{ do}$$
$$\quad i \leftarrow i + 1;$$
$$\quad s \xleftarrow{\$} \textbf{Flip};$$
$$\quad ct \leftarrow s \ ? \ ct + 1 : ct$$

(The last line uses the *ternary conditional operator*—$s \ ? \ ct + 1 : ct$ is equal to $ct + 1$ if s is true, otherwise equal to ct.)

We consider two runs of this program executing T_1 and T_2 iterations, where $T_1 \leq T_2$ are logical variables; call the two programs $sdom_1$ and $sdom_2$. By soundness of the logic and Proposition 2.1.14, the distribution of ct in the second run stochastically dominates the distribution of ct in the first run if we can prove the judgment

$$\vdash sdom_1 \sim sdom_2 : \top \Longrightarrow ct\langle 1\rangle \leq ct\langle 2\rangle.$$

Encoding the argument from Example 2.1.17 in PRHL requires a bit of work. The main obstacle is that the two-sided loop rule in PRHL can only analyze loops in a synchronized fashion, but this is not possible here: when $T_1 < T_2$ the two loops run for different numbers of iterations, no matter how we couple the samples. To get around this problem, we use the equivalence rule $[\text{EQUIV}]$ to transform *sdom* into a more convenient form using the following equivalence:

$$\textbf{while } e \textbf{ do } c \equiv \textbf{while } e \wedge e' \textbf{ do } c; \textbf{while } e \textbf{ do } c$$

This transformation, known in the compilers literature as *loop splitting* (Callahan and Kennedy, 1988), separates out the first iterations where e' holds, and then runs the original loop to completion. We transform $sdom_2$ as follows:

$$sdom'_{2a} \triangleq \left\{ \begin{array}{ll} i \leftarrow 0; ct \leftarrow 0; & i \leftarrow 0; ct \leftarrow 0; \\ \textbf{while } i < T_2 \wedge i < T_1 \textbf{ do} & \textbf{while } i < T_1 \textbf{ do} \\ \quad i \leftarrow i + 1; & \quad i \leftarrow i + 1; \\ \quad s \xleftarrow{\$} \textbf{Flip}; & \quad s \xleftarrow{\$} \textbf{Flip}; \\ \quad ct \leftarrow s \ ? \ ct + 1 : ct; & \quad ct \leftarrow s \ ? \ ct + 1 : ct; \end{array} \right\} \triangleq sdom_1$$

$$sdom'_{2b} \triangleq \left\{ \begin{array}{l} \textbf{while } i < T_2 \textbf{ do} \\ \quad i \leftarrow i + 1; \\ \quad s \xleftarrow{\$} \textbf{Flip}; \\ \quad ct \leftarrow s \ ? \ ct + 1 : ct \end{array} \right.$$

We aim to relate $sdom'_{2a}; sdom'_{2b}$ to $sdom_1$. First, we apply the two-sided rule $[\textsc{While}]$ to relate $sdom_1$ to $sdom'_{2a}$. Taking the identity coupling with $f = \mathrm{id}$ in $[\textsc{Sample}]$, we relate the sampling in the loop body via

$$\textsc{Sample} \ \frac{f = \mathrm{id}}{\vdash s \xleftarrow{\$} \mathbf{Flip} \sim s \xleftarrow{\$} \mathbf{Flip} : \top \Longrightarrow s\langle 1 \rangle = s\langle 2 \rangle}$$

and establish the loop invariant

$$\Theta \triangleq i\langle 1 \rangle = i\langle 2 \rangle \wedge ct\langle 1 \rangle = ct\langle 2 \rangle,$$

proving the judgment

$$\vdash sdom_1 \sim sdom'_{2a} : \top \Longrightarrow \Theta.$$

Then we use the one-sided rule $[\textsc{While-R}]$ for the loop $sdom'_{2b}$ with loop invariant $ct\langle 1 \rangle \le ct\langle 2 \rangle$:

$$\vdash \mathbf{skip} \sim sdom'_{2b} : \Theta \Longrightarrow ct\langle 1 \rangle \le ct\langle 2 \rangle.$$

Composing these two judgments with $[\textsc{Seq}]$ and applying $[\textsc{Equiv}]$ gives the desired judgment:

$$\textsc{Equiv} \ \frac{\vdash sdom_1; \mathbf{skip} \sim sdom'_{2a}; sdom'_{2b} : \top \Longrightarrow ct\langle 1 \rangle \le ct\langle 2 \rangle}{\vdash sdom_1 \sim sdom_2 : \top \Longrightarrow ct\langle 1 \rangle \le ct\langle 2 \rangle}$$

using the equivalence $sdom_1; \mathbf{skip} \equiv sdom_1$.

Probabilistic convergence

In our final example, we build a coupling witnessing convergence of two *random walks*. Each process begins at an integer starting point *start*, and proceeds for T steps. At each step it flips a fair coin. If true, it increases the current position by 1; otherwise, it decreases the position by 1. Given two random walks starting at different initial locations, we want to bound the distance between the two resulting output distributions in terms of T. Intuitively, the position distributions spread out as the random walks proceed, tending towards the uniform distribution on the even integers or the uniform distribution over the odd integers depending on the parity of the initial position and the number of steps. If two walks initially have the same parity (i.e., their starting positions differ by an even integer), then their distributions after taking the same number of steps T should approach one another in total variation distance.

We model a single random walk with the following program *rwalk*:

$$
\begin{aligned}
&pos \leftarrow start; i \leftarrow 0; hist \leftarrow [start]; \\
&\mathbf{while}\ i < T\ \mathbf{do} \\
&\quad i \leftarrow i + 1; \\
&\quad r \xleftarrow{\$} \mathbf{Flip}; \\
&\quad pos \leftarrow pos + (r\ ?\ 1 : -1); \\
&\quad hist \leftarrow pos :: hist
\end{aligned}
$$

The last command records the history of the walk in *hist*; this *ghost variable* does not affect the final output value, but will be useful for our assertions.

By Theorem 2.1.16, we can bound the TV-distance between the position distributions by constructing a coupling where the probability of $pos\langle 1 \rangle \neq pos\langle 2 \rangle$ tends to 0 as T increases. We don't have the tools yet to reason about this probability (we will revisit this point in the next chapter), but for now we can build the coupling and prove the judgment

$$\vdash rwalk \sim rwalk : start\langle 2 \rangle - start\langle 1 \rangle = 2K \Longrightarrow K + start\langle 1 \rangle \in hist\langle 1 \rangle \rightarrow pos\langle 1 \rangle = pos\langle 2 \rangle$$

where K is an integer logical variable. The pre-condition states that the initial positions are an even distance apart. To read the post-condition, the predicate $K + start\langle 1 \rangle \in hist\langle 1 \rangle$ holds if and only if the first walk has moved to position $K + start\langle 1 \rangle$ at some time in the past; if this has happened, then the two coupled positions must be equal.

Our coupling mirrors the two walks. Each step, we have the walks make symmetric moves by arranging opposite samples. Once the walks meet, we have the walks match each other by coupling the samples to be equal. In this way, if the first walk reaches $start\langle 1 \rangle + K$, then the second walk must be at $start\langle 2 \rangle - K$ since both walks are coupled to move symmetrically. In this case, the initial condition $start\langle 2 \rangle - start\langle 1 \rangle = 2K$ gives

$$pos\langle 1 \rangle = start\langle 1 \rangle + K = start\langle 2 \rangle - K = pos\langle 2 \rangle$$

so the walks meet and continue to share the same position thereafter. This argument requires the starting positions to be an even distance apart so the positions in the two walks always have the same parity; if the two starting positions are an odd distance apart, then the two distributions after T steps have disjoint support and the coupled walks can never meet.

To formalize this argument in PRHL, we handle the loop with the two-sided rule [WHILE] and invariant

$$\Theta \triangleq \begin{cases} |hist\langle 1 \rangle| > 0 \wedge |hist\langle 2 \rangle| > 0 \\ K + start\langle 1 \rangle \in hist\langle 1 \rangle \rightarrow pos\langle 1 \rangle = pos\langle 2 \rangle \\ K + start\langle 1 \rangle \notin hist\langle 1 \rangle \rightarrow pos\langle 2 \rangle - pos\langle 1 \rangle = 2(K - (hd(hist\langle 1 \rangle) - start\langle 1 \rangle)), \end{cases}$$

where $hd(hist)$ is the first element (the *head*) of the non-empty list $hist$. The last two conditions model the two cases. If the first walk has already visited $K + start\langle 1 \rangle$, the walks have already met under the coupling and they must have the same position. Otherwise, the walks have not met. If $d \triangleq hd(hist\langle 1 \rangle) - start\langle 1 \rangle$ is the (signed) distance the first walk has moved away from its starting location and the two walks are initially $2K$ apart, then the current distance between coupled positions must be $2(K - d)$.

To show the invariant is preserved, we perform a case analysis with [CASE]. If $K + start\langle 1 \rangle \in hist\langle 1 \rangle$ holds then the walks have already met in the past and currently have the same position (by Θ). So, we select the identity coupling in [SAMPLE]:

$$\text{SAMPLE} \ \frac{f = \text{id}}{\vdash r \xleftarrow{\$} \textbf{Flip} \sim r \xleftarrow{\$} \textbf{Flip} : K + start\langle 1 \rangle \in hist\langle 1 \rangle \implies r\langle 1 \rangle = r\langle 2 \rangle}.$$

Since $K + start\langle 1 \rangle \in hist\langle 1 \rangle \rightarrow pos\langle 1 \rangle = pos\langle 2 \rangle$ holds at the start of the loop, we know $pos\langle 1 \rangle = pos\langle 2 \rangle$ at the end of the loop; since $K + start\langle 1 \rangle \in hist\langle 1 \rangle$ is preserved by the loop body, the invariant Θ holds.

Otherwise if $K + start\langle 1 \rangle \notin h\langle 1 \rangle$, then the walks have not yet met and should be mirrored. So, we select the negation coupling with $f = \neg$ in [SAMPLE]:

$$\text{SAMPLE} \ \frac{f = \neg}{\vdash r \xleftarrow{\$} \textbf{Flip} \sim r \xleftarrow{\$} \textbf{Flip} : K + start\langle 1 \rangle \notin hist\langle 1 \rangle \implies \neg r\langle 1 \rangle = r\langle 2 \rangle}$$

To show the loop invariant, there are two cases. If $K + start\langle 1 \rangle \in h\langle 1 \rangle$ holds after the body, the two walks have just met for the first time and $pos\langle 1 \rangle = pos\langle 2 \rangle$ holds. Otherwise, the walks remain mirrored: $pos\langle 1 \rangle$ increased by $r\langle 1 \rangle$ and $pos\langle 2 \rangle$ decreased by $r\langle 1 \rangle$, so $pos\langle 2 \rangle - pos\langle 1 \rangle = 2(K + (hd(hist\langle 1 \rangle) - start\langle 1 \rangle))$ and the invariant Θ is preserved.

Putting it all together, we have the desired judgment:

$$\vdash rwalk \sim rwalk : start\langle 2 \rangle - start\langle 1 \rangle = 2K \implies K + start\langle 1 \rangle \in h\langle 1 \rangle \rightarrow pos\langle 1 \rangle = pos\langle 2 \rangle.$$

While this judgment describes a coupling between the position distributions, we need to analyze finer properties of the coupling distribution to apply Theorem 2.1.16—namely, we must bound the probability that $pos\langle 1 \rangle$ is not equal to $pos\langle 2 \rangle$. We will consider how to extract this information in the next chapter.

2.4   Related work

Relational Hoare logics and probabilistic couplings have been extensively studied in disparate research communities.

Relational Hoare logics

The logic PRHL is a prime example of a *relational* program logic, which extend standard Floyd-Hoare logics to prove properties about two programs. Benton (2004) first designed a relational version of Hoare logic called RHL to prove equivalence between two (deterministic) programs. Benton used his logic to verify compiler transformations, showing the original program is equivalent to the transformed program. Relational versions of other program logics have also been considered, including an extension of separation logic by Yang (2007) to prove relational properties of pointer-manipulating programs. There is nothing particularly special about relating exactly *two* programs; recently, Sousa and Dillig (2016) give a Hoare logic for proving properties of k executions of the same program for arbitrary k.

Barthe et al. (2009) extended Benton's work to prove relational properties of probabilistic programs, leading to the logic PRHL. As we have seen, the key technical insight is to interpret the relational post-condition as a probabilistic lifting between two output distributions. Barthe et al. (2009) used PRHL to verify security properties for a variety of cryptographic protocols by mimicking the so-called *game-hopping* proof technique (Bellare and Rogaway, 2006; Shoup, 2004), where the original program is transformed step-by-step to an obviously secure version (e.g., a program returning a random number). Security follows if each transformation approximately preserves the program semantics. Our analysis of the XOR cipher is a very simple example of this technique; more sophisticated proofs chain together dozens of transformations.

Probabilistic couplings and liftings

Couplings are a well-studied tool in probability theory; readers can consult the lecture notes by Lindvall (2002) or the textbooks by Thorisson (2000) and Levin et al. (2009) for entry points into this vast literature.

Probabilistic liftings were initially proposed in research on *bisimulation*, techniques for proving equivalence of transition systems. Larsen and Skou (1991) were the first to consider a probabilistic notion of bisimulation. Roughly speaking, their definition considers an equivalence relation E on states and requires that any two states in the same equivalence class have the same probability of stepping to any other equivalence class. The construction for arbitrary relations arose soon after, when researchers generalized probabilistic bisimulation to probabilistic simulation; Jonsson and Larsen (1991, Definition 4.3) proposes a *satisfaction relation* using witness distributions, similar to the definition used in PRHL. Desharnais (1999, Definition 3.6.2) and Segala and Lynch (1995, Definition 12) give an alternative characterization without witness distributions, similar to Strassen's theorem (Strassen, 1965); Desharnais (1999, Theorem 7.3.4) observed that both definitions are equivalent in the finite case via the max flow-min cut theorem. Probabilistic (bi)simulation can be characterized logically, i.e., two systems are (bi)similar if and only if they satisfy the same formulas in some modal logic (Desharnais, Edalat, and Panangaden, 2002; Desharnais, Gupta, Jagadeesan, and Panangaden, 2003; Fijalkow, Klin, and Panangaden, 2017; Larsen and Skou, 1991). Deng and Du (2011) survey logical, metric, and algorithmic characterizations of these relations.

Probabilistic liftings have proven to be a convenient abstraction for many styles of formal reasoning beyond bisimulation and program logics. For instance, Barthe, Fournet, Grégoire, Strub, Swamy, and Zanella-Béguelin (2014a) combine probabilistic lifting with a probability monad to prove relational properties in RF*, a refinement type system for a probabilistic, functional language.

Chapter 3

From coupling proofs to product programs

As we have seen, valid judgments in PRHL imply a coupling of two output distributions with a particular support. Some applications of proof by coupling need more detailed information to conclude a relational property; notable examples include coupling proofs for convergence, like the random walk example from the previous chapter. While a valid judgment gives no further information beyond the support of the coupling, we usually have more information at hand—often, we have a proof using the logical rules in PRHL. Since proof rules correspond to steps in proofs by coupling, which indirectly construct a coupling distribution, the structure of PRHL proofs should somehow encode the coupling.

Indeed, this is the case. While we cannot hope to explicitly list the probabilities of every pair under a coupling—for one thing, there may be infinitely many—we show that every PRHL derivation encodes a probabilistic program *generating* the witness. Intuitively, a coupling proof describes how to simulate two probabilistic processes as one, by sharing randomness. Accordingly, proofs in PRHL encode how to combine two programs into one; the witness of a coupling is just the output distribution of the combined program. This construction, which we call the *coupled product*, draws a correspondence between coupling proofs and probabilistic product programs, recalling a theme in computer science and logic: proofs can be viewed as programs.

To make our ideas concrete, we design an extension of PRHL called ×PRHL (*product* PRHL), where judgments construct a coupled product program. Since this program depends on the whole proof derivation and not just the final judgment, there may be multiple ×PRHL judgments corresponding to a given PRHL judgment. We first present a core version of ×PRHL with logical rules based on PRHL (Section 3.1), followed by a novel loop rule that allows asynchronous reasoning (Section 3.2). After establishing soundness (Section 3.3), we apply our logic to prove convergence and rapid mixing for probabilistic processes (Section 3.4), modeling examples of shift couplings (Section 3.5) and path couplings (Section 3.6). Finally, we compare the coupled product to prior constructions (Section 3.7).

3.1 The core logic ×PRHL

The logic ×PRHL extends PRHL by pairing each judgment with a product program.

Judgments and validity

Judgments in ×PRHL have the following form:

$$\left\{ \Phi \right\} \begin{array}{c} c_1 \\ c_2 \end{array} \left\{ \Psi \right\} \blacktriangleright c_\times$$

Just like in PRHL, c_1 and c_2 are probabilistic programs and the pre- and post-conditions Φ and Ψ are assertions on product memories. The new component is the *coupled product* $c_\times$, which simulates two correlated executions of c_1

and c_2. To ensure the two executions do not interfere with one another, $c_\times$ operates on a product memory with two copies of each variable, tagged with $\langle 1 \rangle$ and $\langle 2 \rangle$.

Semantic validity in $\times$PRHL is very similar to validity in PRHL: the output distribution of the product program on two related inputs couples the output distributions of the two given programs.

Definition 3.1.1. Suppose c_1, c_2 have variables in $\mathcal{X} \cup \mathcal{L}$, Φ and Ψ are predicates over $\mathcal{X}\langle 1 \rangle \cup \mathcal{X}\langle 2 \rangle \cup \mathcal{L}$, and $c_\times$ has variables in $\mathcal{X}\langle 1 \rangle \cup \mathcal{X}\langle 2 \rangle \cup \mathcal{L}$. An $\times$PRHL judgment is *valid* in a logical context ρ, written

$$\rho \models \left\{ \Phi \right\} \; \begin{matrix} c_1 \\ c_2 \end{matrix} \; \left\{ \Psi \right\} \blacktriangleright c_\times,$$

if for every two memories $(m_1, m_2) \in \llbracket \Phi \rrbracket_\rho$ we have

1. $\mathrm{supp}(\llbracket c_\times \rrbracket_\rho(m_1, m_2)) \subseteq \llbracket \Psi \rrbracket_\rho$;

2. $\llbracket c_1 \rrbracket_\rho m_1 = \pi_1(\llbracket c_\times \rrbracket_\rho(m_1, m_2))$; and

3. $\llbracket c_2 \rrbracket_\rho m_2 = \pi_2(\llbracket c_\times \rrbracket_\rho(m_1, m_2))$.

(Recall π_1, π_2 are the first and second projections from **SDistr(State$_\times$)** to **SDistr(State)**.)

Core proof rules

Proof rules in $\times$PRHL describe how to construct product programs. Like their PRHL counterparts, the core rules of $\times$PRHL can be divided into three groups: two-sided rules, one-sided rules, and structural rules.

The two-sided rules are presented in Fig. 3.1. For the first rule [SKIP], since the two programs don't have any effect, the coupled program also has no effect. The next pair of rules handle assignment and sampling statements. The rule [ASSN] relates two assignment statements; the product program simply performs both operations on the product memory. The rule [SAMPLE] for random sampling is more interesting. Just like its counterpart in PRHL, this rule is parameterized by a bijection f between the supports of the two distributions. The product program draws the first sample for $x_1\langle 1 \rangle$ from d_1 and then assigns $x_2\langle 2 \rangle$ *deterministically* with $f(x_1\langle 1 \rangle)$—this is the sample corresponding to $x_1\langle 1 \rangle$ under the coupling. In this way, the product program simulates two random draws with a single source of randomness.

The sequential composition rule [SEQ] relates two sequencing commands. The product program is simply the sequential composition of the product programs for the first and second commands, highlighting the compositional nature of couplings.

The final pair of rules relate branching commands. Just like in PRHL, the pre-condition must ensure that the guards are equal. In the rule [COND], the premises give two product programs c and c' relating the two true branches and the two false branches, respectively. The product program for the conditional first branches on the guard and then executes the product program for the corresponding branch. In the rule [WHILE], the product program for the loop executes the product program for the body while the guard remains true.

Next we consider the one-sided proof rules in Fig. 3.2. The first four rules for assignment and sampling, [ASSN-L]/[ASSN-R] and [SAMPLE-L]/[SAMPLE-R], relate a command with **skip**; the product program simply executes the assignment or sampling command on the indicated side.

The one-sided rules for conditionals, [COND-L] and [COND-R], relate a conditional to an arbitrary command (c_2 and c_1, respectively). The premises give two product programs relating the general command with the true and false branches of the conditional. The coupled product branches on the guard—$e_1\langle 1 \rangle$ or $e_2\langle 2 \rangle$—and runs the product program for the corresponding branch.

The one-sided rules for loops, [WHILE-L] and [WHILE-R], are similar. The premises give a product program relating the body of the loop to **skip**; the resulting product program for the loop executes the product program for the body while the loop guard is true. Like the analogous rules in PRHL, the loop must be lossless.

$$\textsc{Skip}\ \frac{}{\vdash \left\{\Phi\right\}\ \begin{matrix}\mathbf{skip}\\ \mathbf{skip}\end{matrix}\ \left\{\Phi\right\}\ \blacktriangleright \mathbf{skip}}$$

$$\textsc{Assn}\ \frac{}{\vdash \left\{\Psi\left\{e_1\langle 1\rangle, e_2\langle 2\rangle / x_1\langle 1\rangle, x_2\langle 2\rangle\right\}\right\}\ \begin{matrix}x_1 \leftarrow e_1\\ x_2 \leftarrow e_2\end{matrix}\ \left\{\Psi\right\}\ \blacktriangleright\ \begin{matrix}x_1\langle 1\rangle \leftarrow e_1\langle 1\rangle;\\ x_2\langle 2\rangle \leftarrow e_2\langle 2\rangle\end{matrix}}$$

$$\textsc{Sample}\ \frac{f : \operatorname{supp}(d_1) \to \operatorname{supp}(d_2)\ \text{bijection}}{\vdash \left\{\forall v \in \operatorname{supp}(d_1),\ \Psi\left\{v, f(v)/x_1\langle 1\rangle, x_2\langle 2\rangle\right\}\right\}\ \begin{matrix}x_1 \overset{\$}{\leftarrow} d_1\\ x_2 \overset{\$}{\leftarrow} d_2\end{matrix}\ \left\{\Psi\right\}\ \blacktriangleright\ \begin{matrix}x_1\langle 1\rangle \overset{\$}{\leftarrow} d_1;\\ x_2\langle 2\rangle \leftarrow f(x_1\langle 1\rangle)\end{matrix}}$$

$$\textsc{Seq}\ \frac{\vdash \left\{\Phi\right\}\ \begin{matrix}c_1\\ c_2\end{matrix}\ \left\{\Psi\right\}\ \blacktriangleright c \qquad \vdash \left\{\Psi\right\}\ \begin{matrix}c'_1\\ c'_2\end{matrix}\ \left\{\Theta\right\}\ \blacktriangleright c'}{\vdash \left\{\Phi\right\}\ \begin{matrix}c_1; c'_1\\ c_2; c'_2\end{matrix}\ \left\{\Theta\right\}\ \blacktriangleright c; c'}$$

$$\textsc{Cond}\ \frac{\models \Phi \to e_1\langle 1\rangle = e_2\langle 2\rangle \qquad \vdash \left\{\Phi \wedge e_1\langle 1\rangle\right\}\ \begin{matrix}c_1\\ c_2\end{matrix}\ \left\{\Psi\right\}\ \blacktriangleright c \qquad \vdash \left\{\Phi \wedge \neg e_1\langle 1\rangle\right\}\ \begin{matrix}c'_1\\ c'_2\end{matrix}\ \left\{\Psi\right\}\ \blacktriangleright c'}{\vdash \left\{\Phi\right\}\ \begin{matrix}\mathbf{if}\ e_1\ \mathbf{then}\ c_1\ \mathbf{else}\ c'_1\\ \mathbf{if}\ e_2\ \mathbf{then}\ c_2\ \mathbf{else}\ c'_2\end{matrix}\ \left\{\Psi\right\}\ \blacktriangleright \mathbf{if}\ e_1\langle 1\rangle\ \mathbf{then}\ c\ \mathbf{else}\ c'}$$

$$\textsc{While}\ \frac{\models \Phi \to e_1\langle 1\rangle = e_2\langle 2\rangle \qquad \vdash \left\{\Phi \wedge e_1\langle 1\rangle\right\}\ \begin{matrix}c_1\\ c_2\end{matrix}\ \left\{\Phi\right\}\ \blacktriangleright c}{\vdash \left\{\Phi\right\}\ \begin{matrix}\mathbf{while}\ e_1\ \mathbf{do}\ c_1\\ \mathbf{while}\ e_2\ \mathbf{do}\ c_2\end{matrix}\ \left\{\Phi \wedge \neg e_1\langle 1\rangle\right\}\ \blacktriangleright \mathbf{while}\ e_1\langle 1\rangle\ \mathbf{do}\ c}$$

Figure 3.1: Two-sided $\times$PRHL rules

Finally, we come to the structural rules in Fig. 3.3. The rules [CONSEQ] and [EQUIV] are straightforward: the former rule preserves the product program of the premise, while the latter rule replaces programs by equivalent programs. The rule [CASE] is more interesting; recall that this rule performs a case analysis on the two input memories. The product programs from the two logical cases are combined into a final product program that branches on the predicate and selects the corresponding product program. In this way, a logical case analysis is realized by a branching statement in the product program. Unlike in PRHL, this rule performs a case analysis on an *expression e* instead of a general predicate Θ in the assertion logic; this restriction is needed to reflect the predicate as a guard expression in the product.[1] Finally, [FRAME] is the $\times$PRHL version of the frame rule.

Remark 3.1.2. The careful reader may notice that we do not give an analogous rule for the transitivity rule [TRANS] from PRHL. Given two product programs for the premises, it is not clear how to construct a product program for the conclusion; intuitively, we want to somehow interleave the product programs together while carefully aligning samples. Finding a $\times$PRHL version of this rule is an interesting open problem.

---

[1]For instance, there is no boolean expression corresponding to universal or existential quantification; such an expression would typically not be computable.

$$\text{ASSN-L} \frac{}{\vdash \left\{ \Psi\{e_1\langle 1\rangle / x_1\langle 1\rangle\} \right\} \begin{array}{c} x_1 \leftarrow e_1 \\ \textbf{skip} \end{array} \left\{ \Psi \right\} \blacktriangleright x_1\langle 1\rangle \leftarrow e_1\langle 1\rangle}$$

$$\text{ASSN-R} \frac{}{\vdash \left\{ \Psi\{e_2\langle 2\rangle / x_2\langle 2\rangle\} \right\} \begin{array}{c} \textbf{skip} \\ x_2 \leftarrow e_2 \end{array} \left\{ \Psi \right\} \blacktriangleright x_2\langle 2\rangle \leftarrow e_2\langle 2\rangle}$$

$$\text{SAMPLE-L} \frac{}{\vdash \left\{ \forall v \in \text{supp}(d_1),\ \Psi\{v/x_1\langle 1\rangle\} \right\} \begin{array}{c} x_1 \xleftarrow{\$} d_1 \\ \textbf{skip} \end{array} \left\{ \Psi \right\} \blacktriangleright x_1\langle 1\rangle \xleftarrow{\$} d_1}$$

$$\text{SAMPLE-R} \frac{}{\vdash \left\{ \forall v \in \text{supp}(d_2),\ \Psi\{v/x_2\langle 2\rangle\} \right\} \begin{array}{c} \textbf{skip} \\ x_2 \xleftarrow{\$} d_2 \end{array} \left\{ \Psi \right\} \blacktriangleright x_2\langle 2\rangle \xleftarrow{\$} d_2}$$

$$\text{COND-L} \frac{\vdash \left\{ \Phi \wedge e_1\langle 1\rangle \right\} \begin{array}{c} c_1 \\ c_2 \end{array} \left\{ \Psi \right\} \blacktriangleright c \quad \vdash \left\{ \Phi \wedge \neg e_1\langle 1\rangle \right\} \begin{array}{c} c_1' \\ c_2 \end{array} \left\{ \Psi \right\} \blacktriangleright c'}{\vdash \left\{ \Phi \right\} \begin{array}{c} \textbf{if } e_1 \textbf{ then } c_1 \textbf{ else } c_1' \\ c_2 \end{array} \left\{ \Psi \right\} \blacktriangleright \textbf{if } e_1\langle 1\rangle \textbf{ then } c \textbf{ else } c'}$$

$$\text{COND-R} \frac{\vdash \left\{ \Phi \wedge e_2\langle 2\rangle \right\} \begin{array}{c} c_1 \\ c_2 \end{array} \left\{ \Psi \right\} \blacktriangleright c \quad \vdash \left\{ \Phi \wedge \neg e_2\langle 2\rangle \right\} \begin{array}{c} c_1 \\ c_2' \end{array} \left\{ \Psi \right\} \blacktriangleright c'}{\vdash \left\{ \Phi \right\} \begin{array}{c} c_1 \\ \textbf{if } e_2 \textbf{ then } c_2 \textbf{ else } c_2' \end{array} \left\{ \Psi \right\} \blacktriangleright \textbf{if } e_2\langle 2\rangle \textbf{ then } c \textbf{ else } c'}$$

$$\text{WHILE-L} \frac{\vdash \left\{ \Phi \wedge e_1\langle 1\rangle \right\} \begin{array}{c} c_1 \\ \textbf{skip} \end{array} \left\{ \Phi \right\} \blacktriangleright c \quad \models \Phi \to \Phi_1\langle 1\rangle \quad \Phi_1 \models \textbf{while } e_1 \textbf{ do } c_1 \text{ lossless}}{\vdash \left\{ \Phi \right\} \begin{array}{c} \textbf{while } e_1 \textbf{ do } c_1 \\ \textbf{skip} \end{array} \left\{ \Phi \wedge \neg e_1\langle 1\rangle \right\} \blacktriangleright \textbf{while } e_1\langle 1\rangle \textbf{ do } c}$$

$$\text{WHILE-R} \frac{\vdash \left\{ \Phi \wedge e_2\langle 2\rangle \right\} \begin{array}{c} \textbf{skip} \\ c_2 \end{array} \left\{ \Phi \right\} \blacktriangleright c \quad \models \Phi \to \Phi_2\langle 1\rangle \quad \Phi_2 \models \textbf{while } e_2 \textbf{ do } c_2 \text{ lossless}}{\vdash \left\{ \Phi \right\} \begin{array}{c} \textbf{skip} \\ \textbf{while } e_2 \textbf{ do } c_2 \end{array} \left\{ \Phi \wedge \neg e_2\langle 2\rangle \right\} \blacktriangleright \textbf{while } e_2\langle 2\rangle \textbf{ do } c}$$

Figure 3.2: One-sided $\times$PRHL rules

$$\textsc{Conseq}\ \dfrac{\vdash \left\{\Phi'\right\}\ \begin{matrix} c_1 \\ c_2 \end{matrix}\ \left\{\Psi'\right\} \blacktriangleright c \qquad \models \Phi \to \Phi' \qquad \models \Psi' \to \Psi}{\vdash \left\{\Phi\right\}\ \begin{matrix} c_1 \\ c_2 \end{matrix}\ \left\{\Psi\right\} \blacktriangleright c}$$

$$\textsc{Equiv}\ \dfrac{\vdash \left\{\Phi\right\}\ \begin{matrix} c_1' \\ c_2' \end{matrix}\ \left\{\Psi\right\} \blacktriangleright c' \qquad c_1 \equiv c_1' \qquad c_2 \equiv c_2' \qquad c \equiv c'}{\vdash \left\{\Phi\right\}\ \begin{matrix} c_1 \\ c_2 \end{matrix}\ \left\{\Psi\right\} \blacktriangleright c}$$

$$\textsc{Case}\ \dfrac{\vdash \left\{\Phi \wedge e\right\}\ \begin{matrix} c_1 \\ c_2 \end{matrix}\ \left\{\Psi\right\} \blacktriangleright c \qquad \vdash \left\{\Phi \wedge \neg e\right\}\ \begin{matrix} c_1 \\ c_2 \end{matrix}\ \left\{\Psi\right\} \blacktriangleright c'}{\vdash \left\{\Phi\right\}\ \begin{matrix} c_1 \\ c_2 \end{matrix}\ \left\{\Psi\right\} \blacktriangleright \textbf{if } e \textbf{ then } c \textbf{ else } c'}$$

$$\textsc{Frame}\ \dfrac{\vdash \left\{\Phi\right\}\ \begin{matrix} c_1 \\ c_2 \end{matrix}\ \left\{\Psi\right\} \blacktriangleright c \qquad \mathrm{FV}(\Theta) \cap \mathrm{MV}(c) = \varnothing}{\vdash \left\{\Phi \wedge \Theta\right\}\ \begin{matrix} c_1 \\ c_2 \end{matrix}\ \left\{\Psi \wedge \Theta\right\} \blacktriangleright c}$$

Figure 3.3: Structural $\times$PRHL rules

$$\textsc{While-Gen}\ \dfrac{\begin{array}{c} \models \Phi \to (e_1\langle 1\rangle \vee e_2\langle 2\rangle) = e \qquad \models \Phi \wedge e \to p_0 \oplus p_1 \oplus p_2 \\[2pt] \models \Phi \wedge p_0 \wedge e \to e_1\langle 1\rangle = e_2\langle 2\rangle \qquad \models \Phi \wedge p_1 \wedge e \to e_1\langle 1\rangle \wedge \Phi_1\langle 1\rangle \qquad \models \Phi \wedge p_2 \wedge e \to e_2\langle 2\rangle \wedge \Phi_2\langle 2\rangle \\[2pt] \Phi_1 \models \textbf{while } e_1 \wedge p_1 \textbf{ do } c_1 \text{ lossless} \qquad \Phi_2 \models \textbf{while } e_2 \wedge p_2 \textbf{ do } c_2 \text{ lossless} \\[2pt] \vdash \left\{\Phi \wedge e \wedge p_0\right\}\ \begin{matrix} (\textbf{if } e_1 \textbf{ then } c_1)^{K_1} \\ (\textbf{if } e_2 \textbf{ then } c_2)^{K_2} \end{matrix}\ \left\{\Phi\right\} \blacktriangleright c_0' \quad \text{with} \quad K_1 > 0, K_2 > 0 \\[2pt] \vdash \left\{\Phi \wedge e_1 \wedge p_1\right\}\ \begin{matrix} c_1 \\ \textbf{skip} \end{matrix}\ \left\{\Phi\right\} \blacktriangleright c_1' \qquad \vdash \left\{\Phi \wedge e_2 \wedge p_2\right\}\ \begin{matrix} \textbf{skip} \\ c_2 \end{matrix}\ \left\{\Phi\right\} \blacktriangleright c_2' \end{array}}{\vdash \left\{\Phi\right\}\ \begin{matrix} \textbf{while } e_1 \textbf{ do } c_1 \\ \textbf{while } e_2 \textbf{ do } c_2 \end{matrix}\ \left\{\Phi \wedge \neg e_1\langle 1\rangle \wedge \neg e_2\langle 2\rangle\right\} \blacktriangleright \begin{matrix} \textbf{while } e \textbf{ do if } p_0 \textbf{ then } c_0' \\ \textbf{else if } p_1 \textbf{ then } c_1' \textbf{ else } c_2' \end{matrix}}$$

Figure 3.4: Asynchronous loop rule [WHILE-GEN] for $\times$PRHL

3.2 An asynchronous loop rule

The logic $\times$PRHL inherits two kinds of loop rules from PRHL. The two-sided rule relates two loops by relating their bodies, a useful principle since the loop bodies are often highly similar. However, this rule requires that the two loops remain synchronized under the coupling. The one-sided loop rules don't require synchronization, but they are significantly weaker—they can only relate a loop to the trivial program **skip**. Taking a slightly broader view, each rule captures one way of analyzing loops: (i) relating a block of iterations in the first with a block of iterations in the second; (ii) relating one iteration in the first with no iterations in the second; and (iii) relating one iteration in the second with no iterations in the first.

To support all three kinds of reasoning, we give a new rule $[\text{WHILE-GEN}]$ in Fig. 3.4. The three analyses can be freely intermixed, resulting in a powerful principle for analyzing loops *asynchronously*. We will step through the premises from top to bottom, starting with the side-conditions. First, we specify an expression e in the product memory that is true if either loop guard is true. Then we specify three boolean flags p_0, p_1, p_2 indicating which of the three cases to apply; exactly one of the flags must be true. The second group of premises ensure the flags and the loop guards are consistent: if p_0 is true, then both guards should be true since we are relating iterations from both loops; if p_1 is true, then the first guard e_1 should be true since we want to relate one iteration in the first loop; if p_2 is true, then the second guard e_2 should be true to relate one iteration in the second loop. The remaining side-conditions guarantee the product programs for the one-sided cases terminate with probability 1; these conditions are needed for soundness. (Intuitively, the one-sided cases can effectively couple **skip** to a loop. This kind of coupling requires losslessness, as we saw in the one-sided loop rules and in Proposition 2.1.9.)

The main $\times$PRHL premises handle the three cases. We write c^K with a constant K for

$$c^K \triangleq \underbrace{c \; ; \cdots ; c}_{K \text{ iterations}}$$

The first $\times$PRHL premise handles the first case: p_0 is true so we relate K_1 iterations of the first loop with K_2 iterations of the second loop, skipping iterations if either loop terminates early. The second and third $\times$PRHL premises handle the second and third cases: p_1 or p_2 is true, and we relate one iteration of the first or second side to **skip**. In the conclusion, the product program interleaves the two original loops depending on the case—it branches on p_0, p_1, p_2, and runs the product program from the corresponding premise.

While we introduce $[\text{WHILE-GEN}]$ for $\times$PRHL, simply dropping the product programs recovers a sound loop rule for PRHL. Some proofs that previously required reasoning outside of the program logic, for instance using program equivalences, can be handled with the extended loop rule. For example, consider the stochastic domination example we first saw in Example 2.1.17 with the program *sdom*:

$$
\begin{aligned}
&i \leftarrow 0; ct \leftarrow 0; \\
&\textbf{while } i < T \textbf{ do} \\
&\quad i \leftarrow i + 1; \\
&\quad s \xleftarrow{\$} \textbf{Flip}; \\
&\quad ct \leftarrow s \; ? \; ct + 1 : ct
\end{aligned}
$$

and recall we considered two versions of this program, $sdom_1$ and $sdom_2$, where the number of iterations was T_1 and T_2 respectively with $T_1 \leq T_2$. When we previously proved the judgment

$$\vdash sdom_1 \sim sdom_2 : \top \implies ct\langle 1 \rangle \leq ct\langle 2 \rangle,$$

showing stochastic domination, we crucially used the program equivalence rule $[\text{EQUIV}]$ to split the loop in $sdom_2$ into two pieces, using the two-sided rule $[\text{WHILE}]$ to analyze the first piece and the one-sided rule $[\text{WHILE-R}]$ to analyze the second piece. The PRHL version of the general rule $[\text{WHILE-GEN}]$ subsumes both loop rules, allowing us to freely switch between two-sided and one-sided reasoning. As a result, we can prove the desired judgment without transforming the programs by using $[\text{WHILE-GEN}]$, with parameters

$$
\begin{aligned}
K_1, K_2 &\triangleq 1 \\
p_0 &\triangleq i\langle 1 \rangle < T_1 \\
p_1 &\triangleq \textbf{false} \\
p_2 &\triangleq T_1 \leq i\langle 2 \rangle < T_2 \\
\Phi &\triangleq (i\langle 1 \rangle < T_1 \rightarrow i\langle 1 \rangle = i\langle 2 \rangle) \wedge ct\langle 1 \rangle \leq ct\langle 2 \rangle.
\end{aligned}
$$

When the first guard p_0 is true, both loops have not terminated and we can analyze the bodies synchronously. The second guard p_1 is always false since we never want to skip iterations on the second side, while the third guard

p_2 is true once the first program has terminated—in this case, we advance the second program alone. We take the couplings from before: the identity coupling in [SAMPLE] when p_0 is true, and the one-sided rule [SAMPLE-R] when p_2 is true.

3.3   Soundness of the logic

The full proof system of ×PRHL is sound.

Theorem 3.3.1 (Soundness of ×PRHL). *Let ρ be a logical context. If a judgment is derivable*

$$\rho \vdash \left\{ \Phi \right\} \begin{array}{c} c_1 \\ c_2 \end{array} \left\{ \Psi \right\} \blacktriangleright c_\times,$$

then it is valid:

$$\rho \models \left\{ \Phi \right\} \begin{array}{c} c_1 \\ c_2 \end{array} \left\{ \Psi \right\} \blacktriangleright c_\times.$$

Proof sketch. By induction on the derivation, performing a case analysis on the final rule. Most of the cases are straightforward. The most complex case, by far, handles the asynchronous rule [WHILE-GEN]. While we can derive the other loop rules (the two-sided rule [WHILE] and the one-sided rules [WHILE-L]/[WHILE-R]) from [WHILE-GEN] and some basic program equivalences, we consider the simpler loop rules as separate cases to decompose the proof for [WHILE-GEN] as much as possible. We present the detailed proof in Appendix A.   □

The natural counterpart to soundness is *completeness*: valid judgments should be derivable in the proof system. It is possible to show ×PRHL is complete in a certain sense for deterministic programs,[2] but currently very little is known about probabilistic programs. We return to this point in Chapter 6.

3.4   Proving probabilistic convergence

The coupled product generates the coupling in a ×PRHL judgment. By analyzing the product program, we can bound the probability of specific events in the coupling distribution to prove quantitative probabilistic relational properties. To demonstrate, we construct couplings in ×PRHL for proving convergence bounds for probabilistic processes, using standard coupling arguments and more advanced variants like shift coupling and path coupling. In each case, we first build the coupling as an ×PRHL judgment and then analyze the coupled product.

Our main goal in this section is to demonstrate the product construction and to show how it mirrors the corresponding informal proof by coupling. While constructing the coupling and generating the coupled product are easily handled by ×PRHL, formally reasoning about the product program is more difficult. The target properties are probabilistic and non-relational, beyond the reach of ×PRHL. To keep the exposition as light as possible, we will sketch our proofs about the coupled product in a standard mathematical style instead of introducing a separate formal system (e.g., PPDL (Kozen, 1985) or PGCL (Morgan, McIver, and Seidel, 1996)). General-purpose theorem provers (such as COQ or AGDA) should also be able to prove the required properties after formalizing enough of probability theory, but such an approach would be quite heavy. Developing more lightweight, easier-to-use techniques for probabilistic non-relational properties remains a significant open challenge.

---

[2]More formally, *relatively complete* for terminating programs given basic equivalences like $c \equiv c; \mathbf{skip}$.

We begin by revisiting the simple random walk program *rwalk* from Section 2.3:

$$pos \leftarrow start; i \leftarrow 0; hist \leftarrow [start];$$
$$\textbf{while } i < T \textbf{ do}$$
$$\quad i \leftarrow i + 1;$$
$$\quad r \xleftarrow{\$} \textbf{Flip};$$
$$\quad pos \leftarrow pos + (r \ ? \ 1 : -1);$$
$$\quad hist \leftarrow pos :: hist$$

Previously, we proved the following judgment in PRHL:

$$\vdash rwalk \sim rwalk : start\langle 2 \rangle - start\langle 1 \rangle = 2K \implies K + start\langle 1 \rangle \in hist\langle 1 \rangle \rightarrow pos\langle 1 \rangle = pos\langle 2 \rangle.$$

The two walks are initially $2K$ apart and the predicate $K + start\langle 1 \rangle \in hist\langle 1 \rangle$ is true exactly when the walks have met under the coupling. Replaying the proof using the corresponding $\times$PRHL rules yields

$$\vdash \left\{ start\langle 2 \rangle - start\langle 1 \rangle = 2K \right\} \begin{array}{c} rwalk \\ rwalk \end{array} \left\{ K + start\langle 1 \rangle \in hist\langle 1 \rangle \rightarrow pos\langle 1 \rangle = pos\langle 2 \rangle \right\} \blacktriangleright rwalk_\times, \qquad (3.1)$$

where $rwalk_\times$ is the following product program:

$$pos\langle 1 \rangle \leftarrow start\langle 1 \rangle; pos\langle 2 \rangle \leftarrow start\langle 2 \rangle;$$
$$i\langle 1 \rangle \leftarrow 0; i\langle 2 \rangle \leftarrow 0;$$
$$hist\langle 1 \rangle \leftarrow [start\langle 1 \rangle]; hist\langle 2 \rangle \leftarrow [start\langle 2 \rangle];$$
$$\textbf{while } i\langle 1 \rangle < T \textbf{ do}$$
$$\quad i\langle 1 \rangle \leftarrow i\langle 1 \rangle + 1; i\langle 2 \rangle \leftarrow i\langle 2 \rangle + 1;$$
$$\quad \textbf{if } pos\langle 1 \rangle = pos\langle 2 \rangle \textbf{ then}$$
$$\quad\quad r\langle 1 \rangle \xleftarrow{\$} \textbf{Flip}; r\langle 2 \rangle \leftarrow r\langle 1 \rangle;$$
$$\quad\quad pos\langle 1 \rangle \leftarrow pos\langle 1 \rangle + (r\langle 1 \rangle \ ? \ 1 : -1);$$
$$\quad\quad pos\langle 2 \rangle \leftarrow pos\langle 2 \rangle + (r\langle 2 \rangle \ ? \ 1 : -1);$$
$$\quad\quad hist\langle 1 \rangle \leftarrow pos\langle 1 \rangle :: hist\langle 1 \rangle; hist\langle 2 \rangle \leftarrow pos\langle 2 \rangle :: hist\langle 2 \rangle$$
$$\quad \textbf{else}$$
$$\quad\quad r\langle 1 \rangle \xleftarrow{\$} \textbf{Flip}; r\langle 2 \rangle \leftarrow \neg r\langle 1 \rangle;$$
$$\quad\quad pos\langle 1 \rangle \leftarrow pos\langle 1 \rangle + (r\langle 1 \rangle \ ? \ 1 : -1);$$
$$\quad\quad pos\langle 2 \rangle \leftarrow pos\langle 2 \rangle + (r\langle 2 \rangle \ ? \ 1 : -1);$$
$$\quad\quad hist\langle 1 \rangle \leftarrow pos\langle 1 \rangle :: hist\langle 1 \rangle; hist\langle 2 \rangle \leftarrow pos\langle 2 \rangle :: hist\langle 2 \rangle$$

The structure of the coupled product reflects the coupling proof. For instance, the loop is introduced by the two-sided rule [WHILE], and the conditional statement is introduced by the case analysis [CASE]. Intuitively, this program simulates two coupled random walks. Each iteration, the program branches on whether the positions of the two walks are equal or not, setting the two samples $r\langle 1 \rangle$ and $r\langle 2 \rangle$ to be equal if so, and opposite if not. Thus the positions $pos\langle 1 \rangle$ and $pos\langle 2 \rangle$ trace out two mirrored walks when the positions are different, and a single walk once the positions coincide.

Now, we can bound the distance between the position distributions in the original walks by bounding the probability of $K + start\langle 1 \rangle \notin hist\langle 1 \rangle$ in $rwalk_\times$. We need a basic result from the theory of random walks.

Theorem 3.4.1 (see, e.g., Levin et al. (2009, Theorem 2.17)). *Let $X_0, X_1, \ldots$ be the positions of a simple random walk on the integers starting at $X_0 = q \in \mathbb{Z}$. The probability the walk does not reach 0 within t steps is at most*

$$\Pr[X_0, \ldots, X_t \neq 0] \leq \frac{12q}{\sqrt{t}}.$$

Now we bound the rate of convergence of two random walks.

Theorem 3.4.2. *Let m_1, m_2 be two memories such that $m_2(start) - m_1(start) = 2K$ for $K \in \mathbb{Z}$. Let μ_1, μ_2 be the final distributions over memories:*

$$\mu_1 \triangleq [\![rwalk]\!]m_1 \quad and \quad \mu_2 \triangleq [\![rwalk]\!]m_2.$$

Let η_1, η_2 be the final distributions over positions:

$$\eta_1 \triangleq [\![pos]\!]^{\sharp}(\mu_1) \quad and \quad \eta_2 \triangleq [\![pos]\!]^{\sharp}(\mu_2).$$

Then the distance between the two output distributions over positions is at most

$$d_{tv}(\eta_1, \eta_2) \leq \frac{12K}{\sqrt{T}}.$$

Proof. The basic idea is to analyze the coupled product in the ×PRHL judgment Eq. (3.1) and then apply the coupling method (Theorem 2.1.16), but we need to handle one detail before we can string these results together. The coupling method requires a coupling such that the two samples are *equal* with high probability, but the coupling from the post-condition of Eq. (3.1) only describes when the two *positions* are equal—the coupling is a distribution over pairs of whole memories, which may be different even if the positions are equal.

To address this issue, let $\mu_{\times}$ be the witness in Eq. (3.1) generated by the coupled product and let $\eta_{\times}$ be the projection to the positions:

$$\mu_{\times} \triangleq [\![rwalk_{\times}]\!](m_1, m_2) \quad and \quad \eta_{\times} \triangleq [\![(pos\langle 1\rangle, pos\langle 2\rangle)]\!]^{\sharp}(\mu_{\times}).$$

We directly calculate

$$\Pr_{(p_1, p_2) \sim \eta_{\times}} [p_1 \neq p_2] = \Pr_{(m_1, m_2) \sim \mu_{\times}} [m_1(pos\langle 1\rangle) \neq m_2(pos\langle 2\rangle)]$$
$$\leq \Pr_{(m_1, m_2) \sim \mu_{\times}} [(m_1, m_2) \in [\![K + start\langle 1\rangle \notin hist\langle 1\rangle]\!]],$$

where the inequality follows by the post-condition in Eq. (3.1): pairs of memories satisfying $K + start\langle 1\rangle \in hist\langle 1\rangle$ must have equal positions.

So, it suffices to upper bound the probability of $K + start\langle 1\rangle \notin hist\langle 1\rangle$. Looking at the coupled product $rwalk_{\times}$, as long as the two walks have not met, the *distance* between the two coupled walks behaves like a single random walk: increasing by 2 with probability 1/2, decreasing by 2 with probability 1/2. This derived random walk starts at $start\langle 2\rangle - start\langle 1\rangle = 2K$; if it reaches 0 before T steps, then the two original walks meet and $K + start\langle 1\rangle \in hist\langle 1\rangle$ holds. Accordingly, Theorem 3.4.1 gives

$$\Pr_{(m_1, m_2) \sim \mu_{\times}} [(m_1, m_2) \in [\![K + start\langle 1\rangle \notin hist\langle 1\rangle]\!]] \leq \frac{12K}{\sqrt{T}}$$

so we can conclude

$$d_{tv}(\eta_1, \eta_2) = d_{tv}(\pi_1(\eta_{\times}), \pi_2(\eta_{\times})) \leq \Pr_{(p_1, p_2) \sim \eta_{\times}} [p_1 \neq p_2] \leq \frac{12K}{\sqrt{T}},$$

where the first inequality follows by the coupling method (Theorem 2.1.16). $\qquad \square$

Hence, the distributions approach one another as the number of timesteps T increases.

3.5   Shift couplings

In the previous example, we were able to construct the coupling *synchronously* because the two coupled walks meet at the same iteration. This may not be the case in more complex proofs. To demonstrate, we consider an example of a *shift coupling*—a coupling where the two processes meet at two random timesteps. To construct this kind of coupling, we cannot use the synchronous rule [WHILE] since we may need to relate samples across different iterations. Instead, we will apply our asynchronous rule [WHILE-GEN].

Our example is called the *Dynkin process*.[3] This process maintains a position $pos \in \mathbb{N}$, initialized to $start \in [0,\ldots,10]$. Each step, it draws a uniformly random number r from $[1,\ldots,10]$ and increments the position by r. The process stops as soon as pos exceeds $T \in \mathbb{N}$, returning the final value as the output. The following code implements the Dynkin process:

$$
\begin{aligned}
&pos \leftarrow start; \\
&hist \leftarrow [start]; \\
&\textbf{while } pos < T \textbf{ do} \\
&\quad r \xleftarrow{\$} \textbf{Unif}([1,\ldots,10]); \\
&\quad pos \leftarrow pos + r; \\
&\quad hist \leftarrow pos :: hist
\end{aligned}
$$

We call this program *dynkin* and we write *dynbody* for the loop body. We use a ghost variable *hist* to keep track of the history of visited positions, just like we did for the random walk. We will analyze two executions of *dynkin* starting at different locations and show the distributions over final positions converge as T increases.

Before seeing the proof in ×PRHL, let's first sketch the coupling argument. If the two processes have the same position, then we couple the samplings to return equal values; this keeps the two positions equal. Otherwise, we sample in the process that is behind, temporarily pausing the leading process. Since the sampled process moves at least one step forward in each iteration, the lagging process will overtake (or land on) the leading process in finitely many steps, when we will switch to one of the other cases.

We perform this reasoning in ×PRHL using [WHILE-GEN] with $K_1 = K_2 = 1$. We take the joint guard

$$
e \triangleq (pos\langle 1\rangle < T) \vee (pos\langle 2\rangle < T),
$$

and flags

$$
p_0 \triangleq pos\langle 1\rangle = pos\langle 2\rangle \quad \text{and} \quad p_1 \triangleq pos\langle 1\rangle < pos\langle 2\rangle \quad \text{and} \quad p_2 \triangleq pos\langle 1\rangle > pos\langle 2\rangle.
$$

These cases are clearly mutually exclusive, and one is always true. Furthermore, they satisfy the necessary consistency requirements: $\models p_1 \wedge e \rightarrow (pos\langle 1\rangle < T)$ and $\models p_2 \wedge e \rightarrow (pos\langle 2\rangle < T)$ both hold. Finally, the loops are clearly lossless: the position increases by at least 1 every iteration, so we are in any case for at most T iterations.

With the side-conditions out of the way, we now turn to the main premises. We take the following invariant:

$$
\Theta \triangleq \begin{cases}
|hist\langle 1\rangle| > 0 \wedge |hist\langle 2\rangle| > 0 \\
hist\langle 1\rangle \cap hist\langle 2\rangle \neq \varnothing \rightarrow pos\langle 1\rangle = pos\langle 2\rangle \\
|pos\langle 1\rangle - pos\langle 2\rangle| \leq 10 \\
hd(hist\langle 1\rangle) = pos\langle 1\rangle \wedge hd(hist\langle 2\rangle) = pos\langle 2\rangle \\
\forall t \in tl(hist\langle 2\rangle),\, pos\langle 1\rangle > t \wedge \forall t \in tl(hist\langle 1\rangle),\, pos\langle 2\rangle > t
\end{cases}
$$

Reading from the top, the first line states that the history lists are non-empty. The second conjunct says that if the two processes have visited the same position at some point in the past, then they currently have the same position. The third conjunct states that the coupled positions are at most 10 apart at all times. The fourth line states that the current position is the first element in each history list, and the last two conjuncts state that the position in

---

[3]The name comes from a magic trick, known as *Dynkin's card trick* or *Kruskal's count*.

each process is strictly larger than all the previous positions of the other process; this holds because we always move the lagging process. (We write $tl(hist)$ for the *tail* of a list $hist$, consisting of all but the first element.)

We now prove the three main premises in $[\text{WHILE-GEN}]$.

Premise p_0

When p_0 is true, $pos\langle 1 \rangle = pos\langle 2 \rangle$ and we need to prove

$$\vdash \left\{ \Theta \wedge e \wedge p_0 \right\} \begin{array}{l} \textbf{if } pos < T \textbf{ then } dynbody \\ \textbf{if } pos < T \textbf{ then } dynbody \end{array} \left\{ \Theta \right\} \blacktriangleright dynkin_{\times 0}.$$

Since both guards are true, we use the two-sided rule $[\text{COND}]$. We use $[\text{SAMPLE}]$ with $f = \text{id}$ (the identity coupling), and then the usual assignment rule $[\text{ASSN}]$. The invariant is preserved since p_0 remains true. So, we have the desired judgment with product program $dynkin_{\times 0}$:

$$\begin{array}{l} \textbf{if } pos\langle 1 \rangle < T \textbf{ then} \\ \quad r\langle 1 \rangle \xleftarrow{\$} \textbf{Unif}([1, \ldots, 10]); \\ \quad r\langle 2 \rangle \leftarrow r\langle 1 \rangle; \\ \quad pos\langle 1 \rangle \leftarrow pos\langle 1 \rangle + r\langle 1 \rangle; \\ \quad pos\langle 2 \rangle \leftarrow pos\langle 2 \rangle + r\langle 2 \rangle; \\ \quad hist\langle 1 \rangle \leftarrow pos\langle 1 \rangle :: hist\langle 1 \rangle; \\ \quad hist\langle 2 \rangle \leftarrow pos\langle 2 \rangle :: hist\langle 2 \rangle \end{array}$$

Premise p_1

When p_1 is true, $pos\langle 1 \rangle < pos\langle 2 \rangle$ and we need to prove

$$\vdash \left\{ \Theta \wedge (pos\langle 1 \rangle < T) \wedge p_1 \right\} \begin{array}{l} \textbf{if } pos < T \textbf{ then } dynbody \\ \textbf{skip} \end{array} \left\{ \Theta \right\} \blacktriangleright dynkin_{\times 1}.$$

Since we are relating a program to **skip**, we apply the one-sided rules. To show we preserve Θ, note that $hist\langle 1 \rangle$ and $hist\langle 2 \rangle$ are both non-empty and $hist\langle 1 \rangle \cap hist\langle 2 \rangle$ is initially empty since $pos\langle 1 \rangle < pos\langle 2 \rangle$, so if $hist\langle 1 \rangle \cap hist\langle 2 \rangle \neq \varnothing$ after the loop body then we must have $pos\langle 1 \rangle \in hist\langle 2 \rangle$. The next conjunct $|pos\langle 1 \rangle - pos\langle 2 \rangle| \leq 10$ also holds, since (i) it holds initially, (ii) $pos\langle 1 \rangle < pos\langle 2 \rangle$ initially, and (iii) $pos\langle 1 \rangle$ moves forward by at most 10. The conjuncts involving the head of $hist$ are clear. For the last two conjuncts, $hist\langle 2 \rangle$ is unchanged while $pos\langle 1 \rangle$ increases, so

$$\forall t \in tl(hist\langle 2 \rangle), \; pos\langle 1 \rangle > t$$

continues to hold. Similarly, if $hist\langle 1 \rangle$ is initially $q :: ps$ where q is the initial value of $pos\langle 1 \rangle$, then it ends up being $pos\langle 1 \rangle :: q :: ps$. Since $pos\langle 2 \rangle$ is initially greater than all elements in ps and also greater than q since p_1 holds, we continue to have

$$\forall t \in tl(hist\langle 1 \rangle), \; pos\langle 2 \rangle > t$$

after executing the body. So, we have the desired judgment with the following product program $dynkin_{\times 1}$:

$$\begin{array}{l} \textbf{if } pos\langle 1 \rangle < T \textbf{ then} \\ \quad r\langle 1 \rangle \xleftarrow{\$} \textbf{Unif}([1, \ldots, 10]); \\ \quad pos\langle 1 \rangle \leftarrow pos\langle 1 \rangle + r\langle 1 \rangle; \\ \quad hist\langle 1 \rangle \leftarrow pos\langle 1 \rangle :: hist\langle 1 \rangle \end{array}$$

Premise p_2

This case is nearly identical to the previous case, using the right-side versions instead of left-side versions of the rules. By a symmetric argument, we have

$$\vdash \left\{ \Theta \wedge (p\langle 2 \rangle < T) \wedge p_2 \right\} \; \begin{array}{c} \textbf{skip} \\ \textbf{if } pos < T \textbf{ then } dynbody \end{array} \; \left\{ \Theta \right\} \blacktriangleright dynkin_{\times 2}$$

where $dynkin_{\times 2}$ is the following product program:

$$
\begin{array}{l}
\textbf{if } pos\langle 2 \rangle < T \textbf{ then} \\
\quad r\langle 2 \rangle \xleftarrow{\$} \textbf{Unif}([1, \ldots, 10]); \\
\quad pos\langle 2 \rangle \leftarrow pos\langle 2 \rangle + r\langle 2 \rangle; \\
\quad hist\langle 2 \rangle \leftarrow pos\langle 2 \rangle :: hist\langle 2 \rangle
\end{array}
$$

Putting it all together

Applying [WHILE-GEN], we have the judgment

$$\vdash \left\{ start\langle 1 \rangle, start\langle 2 \rangle \in [1, \ldots, 10] \right\} \; \begin{array}{c} dynkin \\ dynkin \end{array} \; \left\{ hist\langle 1 \rangle \cap hist\langle 2 \rangle \neq \varnothing \to pos\langle 1 \rangle = pos\langle 2 \rangle \right\} \blacktriangleright dynkin_{\times} \qquad (3.2)$$

for the following product program $dynkin_{\times}$:

$$
\begin{array}{l}
pos\langle 1 \rangle \leftarrow start\langle 1 \rangle; pos\langle 2 \rangle \leftarrow start\langle 2 \rangle \\
hist\langle 1 \rangle \leftarrow [start\langle 1 \rangle]; hist\langle 2 \rangle \leftarrow [start\langle 2 \rangle]; \\
\textbf{while } (pos\langle 1 \rangle < T) \vee (pos\langle 2 \rangle < T) \textbf{ do} \\
\quad \textbf{if } pos\langle 1 \rangle = pos\langle 2 \rangle \textbf{ then} \\
\quad\quad \textbf{if } pos\langle 1 \rangle < T \textbf{ then} \\
\quad\quad\quad r\langle 1 \rangle \xleftarrow{\$} \textbf{Unif}([1, \ldots, 10]); \\
\quad\quad\quad r\langle 2 \rangle \leftarrow r\langle 1 \rangle; \\
\quad\quad\quad pos\langle 1 \rangle \leftarrow pos\langle 1 \rangle + r\langle 1 \rangle; pos\langle 2 \rangle \leftarrow pos\langle 2 \rangle + r\langle 2 \rangle; \\
\quad\quad\quad hist\langle 1 \rangle \leftarrow pos\langle 1 \rangle :: hist\langle 1 \rangle; hist\langle 2 \rangle \leftarrow pos\langle 2 \rangle :: hist\langle 2 \rangle \\
\quad \textbf{else if } pos\langle 1 \rangle < pos\langle 2 \rangle \textbf{ then} \\
\quad\quad \textbf{if } pos\langle 1 \rangle < T \textbf{ then} \\
\quad\quad\quad r\langle 1 \rangle \xleftarrow{\$} \textbf{Unif}([1, \ldots, 10]); \\
\quad\quad\quad pos\langle 1 \rangle \leftarrow pos\langle 1 \rangle + r\langle 1 \rangle; \\
\quad\quad\quad hist\langle 1 \rangle \leftarrow pos\langle 1 \rangle :: hist\langle 1 \rangle \\
\quad \textbf{else} \\
\quad\quad \textbf{if } pos\langle 2 \rangle < T \textbf{ then} \\
\quad\quad\quad r\langle 2 \rangle \xleftarrow{\$} \textbf{Unif}([1, \ldots, 10]); \\
\quad\quad\quad pos\langle 2 \rangle \leftarrow pos\langle 2 \rangle + r\langle 2 \rangle; \\
\quad\quad\quad hist\langle 2 \rangle \leftarrow pos\langle 2 \rangle :: hist\langle 2 \rangle
\end{array}
$$

This program models the informal coupling proof: if the positions are equal, we take equal samples and move both processes; otherwise, we move the lagging process while holding the leading process fixed. We can analyze this program to show convergence of two Dynkin processes.

Theorem 3.5.1. *Let m_1, m_2 be two memories such that $m_1(start), m_2(start) \in [0, 10]$. Let μ_1, μ_2 be the final distributions over memories:*

$$\mu_1 \triangleq [\![dynkin]\!] m_1 \quad and \quad \mu_2 \triangleq [\![dynkin]\!] m_2.$$

Let η_1, η_2 be the final distributions over positions:

$$\eta_1 \triangleq [\![pos]\!]^\sharp(\mu_1) \quad and \quad \eta_2 \triangleq [\![pos]\!]^\sharp(\mu_2).$$

Then the distance between the two position distributions is at most

$$d_{tv}(\eta_1, \eta_2) \leq (9/10)^{\lfloor T/10 \rfloor - 1}.$$

Proof. If $T \leq 10$, the claim is trivial. Otherwise, let $\mu_\times$ be the coupling in Eq. (3.2) and let $\eta_\times$ be the coupling projected to the two positions:

$$\mu_\times \triangleq [\![dynkin_\times]\!](m_1, m_2) \quad and \quad \eta_\times \triangleq [\![(pos\langle 1\rangle, pos\langle 2\rangle)]\!]^\sharp(\mu_\times).$$

We directly calculate

$$\Pr_{(p_1, p_2)\sim\eta_\times}[p_1 \neq p_2] = \Pr_{(m_1, m_2)\sim\mu_\times}[m_1(pos) \neq m_2(pos)]$$
$$\leq \Pr_{(m_1, m_2)\sim\mu_\times}[(m_1, m_2) \in [\![hist\langle 1\rangle \cap hist\langle 2\rangle = \varnothing]\!]],$$

where the inequality follows by the post-condition of Eq. (3.2): pairs of memories where $hist\langle 1\rangle \cap hist\langle 2\rangle$ is non-empty do not have different positions.

We turn to the product program to bound the last quantity. If the two process have not met yet, then $hist\langle 1\rangle \cap hist\langle 2\rangle = \varnothing$. Since the processes are at most 10 apart, in each iteration of the loop there is a 9/10 chance the lagging process misses the leading process, preserving $hist\langle 1\rangle \cap hist\langle 2\rangle = \varnothing$. Since both processes move at most 10 positions each iteration, there are at least $\lfloor T/10 \rfloor - 1$ iterations so

$$\Pr_{(m_1, m_2)\sim\mu_\times}[(m_1, m_2) \in [\![hist\langle 1\rangle \cap hist\langle 2\rangle = \varnothing]\!]] \leq (9/10)^{\lfloor T/10 \rfloor - 1}.$$

By the coupling method (Theorem 2.1.16), we conclude

$$d_{tv}(\eta_1, \eta_2) \leq \Pr_{(p_1, p_2)\sim\eta_\times}[p_1 \neq p_2] \leq (9/10)^{\lfloor T/10 \rfloor - 1}. \qquad \square$$

3.6 Path couplings

So far we have used couplings to analyze several *Markov chains*, iterative processes where the state is a randomized function of the previous state. The main state space in our examples has been the integers—the position in the random walk or Dynkin process, or the count of the number of heads in the stochastic domination example. For more complex state spaces it can be unclear how to couple the samplings to guide the two states towards one another, especially if the states are many transitions apart.

To address this issue, Bubley and Dyer (1997) proposed the *path coupling* method, a powerful tool to construct couplings. Before describing their idea, we first set some definitions.

Definition 3.6.1. Let Ω be a finite set of states. We say a metric $d : \Omega \times \Omega \to \mathbb{N}$ is a *path metric* if whenever $d(s, s') > 1$, there exists $s'' \neq s, s'$ such that $d(s, s') = d(s, s'') + d(s'', s')$. We say two states s, s' are *adjacent* if $d(s, s') = 1$. The *diameter* Δ of the state space is the maximum distance between any two states. A *Markov chain* on Ω is defined by iterating a *transition function* $\tau : \Omega \to \mathbf{Distr}(\Omega)$ starting from some initial state.

Then the main theorem of path coupling is as follows.

Theorem 3.6.2 (Bubley and Dyer (1997))**.** *Consider a Markov chain with transition function τ over a state space Ω with path metric d and diameter at most Δ. Suppose for any two adjacent states s and s′, there exists a coupling μ of $\tau(s), \tau(s')$ with*

$$\mathop{\mathbb{E}}_{(r,r')\sim\mu} [d(r,r')] \leq \beta.$$

Let $\mu_1^{(T)}, \mu_2^{(T)}$ be the final distributions from starting in any two states s_1, s_2 and running T steps of the Markov chain. Then there is a coupling μ of $\mu_1^{(T)}, \mu_2^{(T)}$ with

$$d_{tv}\left(\mu_1^{(T)}, \mu_2^{(T)}\right) \leq \mathop{\Pr}_{(r,r')\sim\mu} [r \neq r'] \leq \beta^T \Delta.$$

In particular, the distributions converge in total variation distance exponentially quickly if $\beta < 1$.

Intuitively, path coupling can be seen as a transitivity principle for couplings: if we can couple the distributions after one step from any two adjacent states, then we can extend to a coupling on distributions from any two initial states. While we are not able to internalize this principle in ×PRHL due to the required bounds on expectations, we can still construct and analyze the one-step couplings. (We consider how to handle expected distance bounds and couplings in Chapter 6.) We present two examples from the original paper by Bubley and Dyer (1997).

Glauber dynamics: sampling a proper coloring

The Markov chain in our first example samples approximately uniform graph colorings. It was first analyzed by Jerrum (1995); we follow the subsequent, simpler analysis by Bubley and Dyer (1997) using path coupling. Recall that a finite *graph G* consists of a finite set *V* of *vertices* and a symmetric binary relation *E* relating vertices that are connected by an *edge*; we let $\mathcal{N}_G(v) \subseteq V$ denote the *neighbors* of a vertex *v*, i.e., the set of vertices with an edge to *v*. We write *D* for the *degree* of *G*, i.e., $|\mathcal{N}_G(v)| \leq D$ for all *v*. We write $n \triangleq |V|$ for the number of vertices.

Let *C* be a finite set of *colors*; we write $k \triangleq |C|$ for the number of colors. A *coloring* of *G* is a map $w : V \to C$ assigning a color to each vertex; the state space of our Markov chain will be the set of colorings. Let the path distance *d* on the state space be the number of vertices colored differently under two colorings; evidently, the diameter Δ of this state space is at most the number of vertices *n*. A coloring is *valid* (also called *proper*) if $w(v)$ and $w(v')$ have distinct colors for all $(v, v') \in E$. The following program models *T* steps of the *Glauber dynamics*:

$$
\begin{aligned}
&i \leftarrow 0; \\
&\textbf{while } i < T \textbf{ do } ; \\
&\quad v \xleftarrow{\$} \textbf{Unif}(V); \\
&\quad c \xleftarrow{\$} \textbf{Unif}(C); \\
&\quad \textbf{if } \mathcal{V}_G(w, v, c) \textbf{ then } w \leftarrow w[v \mapsto c]; \\
&\quad i \leftarrow i + 1
\end{aligned}
$$

where the guard $\mathcal{V}_G(w, v, c)$ holds when *c* is valid at *v* in *w*, namely, when there is no neighbor of *v* colored with *c* in *w*. Informally, the algorithm starts from a coloring *w* and iteratively modifies it by uniformly sampling a vertex *v* and a color *c*, recoloring *v* with *c* if it is locally valid. We focus on the loop body, which encodes the transition function of the Markov chain:

$$
\begin{aligned}
&v \xleftarrow{\$} \textbf{Unif}(V); \\
&c \xleftarrow{\$} \textbf{Unif}(C); \\
&\textbf{if } \mathcal{V}_G(w, v, c) \textbf{ then } w \leftarrow w[v \mapsto c]
\end{aligned}
$$

We call this program *glauber*. To apply path coupling (Theorem 3.6.2), we must find a coupling where the expected distance between coupled states is small when $w\langle 1 \rangle$ and $w\langle 2 \rangle$ are initially adjacent.

Theorem 3.6.3. *Let m_1, m_2 be memories with $m_1(w), m_2(w)$ adjacent colorings. Let μ_1, μ_2 be the distributions over memories after running one step of the transition function:*

$$\mu_1 \triangleq [\![glauber]\!]m_1 \quad and \quad \mu_2 \triangleq [\![glauber]\!]m_2.$$

Let η_1, η_2 be the respective distributions over colorings:

$$\eta_1 \triangleq [\![w]\!]^\sharp(\mu_1) \quad and \quad \eta_2 \triangleq [\![w]\!]^\sharp(\mu_2).$$

Then there is a coupling $\eta_\times$ of (η_1, η_2) with

$$\mathop{\mathbb{E}}_{(w_1,w_2)\sim\eta_\times} [d(w_1, w_2)] \leq 1 - 1/n + 2D/kn.$$

If $\eta_1^{(T)}, \eta_2^{(T)}$ are the distributions over the final colorings after T steps starting from any two colorings, then

$$d_{tv}\left(\eta_1^{(T)}, \eta_2^{(T)}\right) \leq (1 - 1/n + 2D/kn)^T \cdot n.$$

Proof. Suppose the initial memories contain adjacent colorings $w\langle 1\rangle$ and $w\langle 2\rangle$. First, we couple the sampling from **Unif**(V) with [SAMPLE], using the identity coupling $f = \mathrm{id}$.

Now notice that the two initial states $w\langle 1\rangle$ and $w\langle 2\rangle$ differ in the color for a single vertex, call it v_0. Letting $a \triangleq w_1(v_0)$ and $b \triangleq w_2(v_0)$, we perform a case analysis on the sampled vertex with [CASE]. If v is a neighbor of the differing vertex v_0, applying [SAMPLE] with the transposition bijection $\pi^{ab} : C \to C$ defined by

$$\pi^{ab}(x) \triangleq \begin{cases} b & : x = a \\ a & : x = b \\ x & : \text{otherwise} \end{cases}$$

ensures $c\langle 2\rangle = \pi^{ab}(c\langle 1\rangle)$. Otherwise, [SAMPLE] with the identity coupling ensures $c\langle 1\rangle = c\langle 2\rangle$. By applying the one-sided rules for conditionals ([COND-L] and [COND-R]) to the left and the right programs, we have

$$\vdash \left\{d(w\langle 1\rangle, w\langle 2\rangle) = 1\right\} \begin{array}{c} glauber \\ glauber \end{array} \left\{d(w\langle 1\rangle, w\langle 2\rangle) \leq 2\right\} \blacktriangleright glauber_\times, \tag{3.3}$$

where $glauber_\times$ is the following product program:

$$
\begin{aligned}
&v\langle 1\rangle \xleftarrow{\$} \mathbf{Unif}(V); v\langle 2\rangle \leftarrow v\langle 1\rangle; \\
&\textbf{if } v\langle 1\rangle \in \mathcal{N}_G(v_0) \textbf{ then} \\
&\quad c\langle 1\rangle \xleftarrow{\$} \mathbf{Unif}(C); c\langle 2\rangle \leftarrow \pi^{ab}(c\langle 1\rangle) \\
&\textbf{else} \\
&\quad c\langle 1\rangle \xleftarrow{\$} \mathbf{Unif}(C); c\langle 2\rangle \leftarrow c\langle 1\rangle \\
&\textbf{if } \mathcal{V}_G(w\langle 1\rangle, v\langle 1\rangle, c\langle 1\rangle) \textbf{ then} \\
&\quad w\langle 1\rangle \leftarrow w\langle 1\rangle[v\langle 1\rangle \mapsto c\langle 1\rangle] \\
&\textbf{if } \mathcal{V}_G(w\langle 2\rangle, v\langle 2\rangle, c\langle 2\rangle) \textbf{ then} \\
&\quad w\langle 2\rangle \leftarrow w\langle 2\rangle[v\langle 2\rangle \mapsto c\langle 2\rangle]
\end{aligned}
$$

We analyze this program to bound the expected distance between states under the coupling. Let the coupling on memories be $\mu_\times \triangleq [\![glauber_\times]\!](m_1, m_2)$, and let the coupling on the final colorings be $\eta_\times \triangleq [\![(w\langle 1\rangle, w\langle 2\rangle)]\!]^\sharp(\mu_\times)$. We have:

$$\mathop{\mathbb{E}}_{(w_1,w_2)\sim\eta_\times} [d] = 0 \cdot \mathop{\Pr}_{(w_1,w_2)\sim\eta_\times} [d = 0] + 1 \cdot \mathop{\Pr}_{(w_1,w_2)\sim\eta_\times} [d = 1] + 2 \cdot \mathop{\Pr}_{(w_1,w_2)\sim\eta_\times} [d = 2]$$

$$\begin{aligned}
&= 1 - \Pr_{(w_1, w_2) \sim \eta_\times}[d = 0] + \Pr_{(w_1, w_2) \sim \eta_\times}[d = 2] \\
&\leq 1 - \Pr_{(m_1, m_2) \sim \mu_\times}[m_1(v) = v_0 \wedge \mathcal{V}_G(m_1(w), v_0, m_1(c))] \\
&\quad + \Pr_{(m_1, m_2) \sim \mu_\times}[m_1(v) \in \mathcal{N}_G(v_0) \wedge m_1(c) = b] \\
&\leq 1 - \frac{1}{n}\left(1 - \frac{D}{k}\right) + \frac{D}{nk} = 1 - \frac{1}{n} + \frac{2D}{nk}.
\end{aligned}$$

The equalities hold because the distance between the resulting colorings is at most two by the post-condition of Eq. (3.3), so $1 = \Pr[d = 0] + \Pr[d = 1] + \Pr[d = 2]$. The first inequality follows since the distance decreases to zero if we select a valid color at v_0, and the distance can only increase to two if we select a neighbor of v_0 and pick the color combination $(c\langle 1\rangle, c\langle 2\rangle) = (b, a)$. The last step follows since each vertex has at most D neighbors, so there are at at least $k - D$ valid colors at any vertex; in particular, the distance decreases to zero if we select v_0 (probability $1/n$) and a valid color (probability at least $1 - D/k$).

Thus, we have constructed a coupling $\eta_\times$ such that

$$\mathbb{E}_{(w_1, w_2) \sim \eta_\times}[d(w_1, w_2)] \leq 1 - 1/n + 2D/kn.$$

By the path coupling theorem (Theorem 3.6.2), we can bound the distance between the T-step distributions $\eta_1^{(T)}, \eta_2^{(T)}$ over w from any two initial colorings:

$$d_{\mathrm{tv}}\left(\eta_1^{(T)}, \eta_2^{(T)}\right) \leq (1 - 1/n + 2D/kn)^T \cdot n.$$

When the number of colors k is at least $2D + 1$, the right-hand side tends to zero exponentially quickly. $\qquad\square$

Remark 3.6.4. Theorem 3.6.3 bounds how fast the Glauber dynamics converges, started from any two colorings. Using basic facts about Markov chains, it is not hard to show that the Glauber dynamics has the uniform distribution over valid colorings of G as a *stationary distribution* (a distribution $\eta \in \mathbf{Distr}(\Omega)$ such that $\mathbf{bind}(\eta, \tau) = \eta$).[4] As a consequence, the Glauber dynamics started in any *valid* coloring converges exponentially quickly to the uniform distribution over valid colorings when $k \geq 2D + 1$. To see this, let μ be the distribution on colorings after T steps started from some valid coloring. Suppose there are M valid colorings on G, and let $\mu_1, \ldots, \mu_M$ be the corresponding distributions over colorings after T steps. Since the uniform distribution η is stationary, we have

$$\eta = \frac{1}{M} \cdot \mu_1 + \cdots + \frac{1}{M} \cdot \mu_M.$$

For every i, Theorem 3.6.3 gives

$$d_{\mathrm{tv}}(\mu, \mu_i) \leq (1 - 1/n + 2D/kn)^T \cdot n.$$

By linearity of TV distance, the output distribution approaches the uniform distribution over valid colorings:

$$d_{\mathrm{tv}}(\mu, \eta) \leq (1 - 1/n + 2D/kn)^T \cdot n.$$

Condensed hard-core lattice gas: sampling an independent set

Our second example is a Markov chain from statistical physics modeling the evolution of a physical system in the *conserved hard-core lattice gas* (CHLG) model (Bubley and Dyer, 1997). Suppose we have a finite set P of *particles*,

---

[4]The Glauber dynamics takes any valid coloring to another valid coloring, and the probability of transitioning from w to w' is equal to the probability of transitioning from w' to w, so the Glauber dynamics is *reversible* over the valid colorings and hence the uniform distribution is stationary.

$s \triangleq |P|$ in total, and we have a finite graph $G = (V, E)$ with degree at most D. A *placement* is a map $w : P \rightarrow V$ placing each particle at a vertex of the graph. We wish to set the particles so that each vertex has at most one particle and no two particles are located at adjacent vertices; we call such a placement *safe*. (In other words, a safe placement is an independent set.)

We analyze a Markov chain to sample a uniformly random safe placement. Take the state space Ω to be the set of all placements (not necessarily safe). The Markov chain starts from an initial placement. Each step, it samples a particle p from P and a vertex v from V uniformly at random, and tries to relocate p to v. If p is safe at v, then the Markov chain updates the placement; otherwise, it leaves the placement unchanged. We model T steps of this dynamics with the following program:

$$
\begin{aligned}
&i \leftarrow 0; \\
&\textbf{while } i < T \textbf{ do }; \\
&\quad p \xleftarrow{\$} \textbf{Unif}(P); \\
&\quad v \xleftarrow{\$} \textbf{Unif}(V); \\
&\quad \textbf{if } \mathcal{S}_G(w, p, v) \textbf{ then } w \leftarrow w[p \mapsto v]; \\
&\quad i \leftarrow i + 1
\end{aligned}
$$

where the guard $\mathcal{S}_G(w, p, v)$ holds when p is valid at v in w, i.e., when there is no other particle located at v or its neighbors. We let the path metric d be the number of particles with different locations under two placements; evidently, the diameter of the state space is at most s. To build a coupling on the one-step distributions from adjacent initial placements, we analyze the transition function *chlg* extracted from the loop body:

$$
\begin{aligned}
&p \xleftarrow{\$} \textbf{Unif}(P); \\
&v \xleftarrow{\$} \textbf{Unif}(V); \\
&\textbf{if } \mathcal{S}_G(w, p, v) \textbf{ then } w \leftarrow w[p \mapsto v]
\end{aligned}
$$

Theorem 3.6.5. *Let m_1, m_2 be memories with $m_1(w), m_2(w)$ adjacent placements. Let μ_1, μ_2 be the respective distributions over memories after running one step of the transition function:*

$$
\mu_1 \triangleq [\![chlg]\!]m_1 \quad and \quad \mu_2 \triangleq [\![chlg]\!]m_2.
$$

Let η_1, η_2 be the respective distributions over placements:

$$
\eta_1 \triangleq [\![w]\!]^\sharp(\mu_1) \quad and \quad \eta_2 \triangleq [\![w]\!]^\sharp(\mu_2).
$$

Then there is a coupling $\eta_\times$ of (η_1, η_2) such that

$$
\mathop{\mathbb{E}}_{(w_1, w_2) \sim \eta_\times} [d(w_1, w_2)] \leq \beta \triangleq \left(1 - \frac{1}{s}\right)\left(1 + \frac{3(D+1)}{n}\right).
$$

If $\eta_1^{(T)}, \eta_2^{(T)}$ are the distributions over final placements after T steps starting from any two placements, then

$$
d_{tv}\left(\eta_1^{(T)}, \eta_2^{(T)}\right) \leq \beta^T \cdot s.
$$

Proof. To couple the two runs we use [SAMPLE] with $f = \text{id}$ twice, ensuring $p\langle 1 \rangle = p\langle 2 \rangle$ and $v\langle 1 \rangle = v\langle 2 \rangle$. Then we apply the one-sided rules for conditionals ([COND-L] and [COND-R]) to the left and the right sides to prove

$$
\vdash \left\{d(w\langle 1\rangle, w\langle 2\rangle) = 1\right\} \begin{array}{c} chlg \\ chlg \end{array} \left\{d(w\langle 1\rangle, w\langle 2\rangle) \leq 2\right\} \blacktriangleright chlg_\times \tag{3.4}
$$

where $chlg_\times$ is the following product program:

$$
\begin{aligned}
&p\langle 1\rangle \xleftarrow{\$} \textbf{Unif}(P); \\
&p\langle 2\rangle \leftarrow p\langle 1\rangle; \\
&v\langle 1\rangle \xleftarrow{\$} \textbf{Unif}(V); \\
&v\langle 2\rangle \leftarrow v\langle 1\rangle; \\
&\textbf{if } \mathcal{S}_G(w\langle 1\rangle, p\langle 1\rangle, v\langle 1\rangle) \textbf{ then} \\
&\quad w\langle 1\rangle \leftarrow w\langle 1\rangle[p\langle 1\rangle \mapsto v\langle 1\rangle] \\
&\textbf{if } \mathcal{S}_G(w\langle 2\rangle, p\langle 2\rangle, v\langle 2\rangle) \textbf{ then} \\
&\quad w\langle 2\rangle \leftarrow w\langle 2\rangle[p\langle 2\rangle \mapsto v\langle 2\rangle]
\end{aligned}
$$

Now we bound the expected distance between the final placements. The two initial placements $w\langle 1\rangle$ and $w\langle 2\rangle$ differ in the position of a single particle p_0, located at vertex a and b in $w\langle 1\rangle$ and $w\langle 2\rangle$ respectively. Let the coupling on output distributions be $\mu_\times \triangleq [\![chlg_\times]\!](m_1, m_2)$ and let the coupling on placement distributions be $\eta_\times \triangleq [\![(w\langle 1\rangle, w\langle 2\rangle)]\!]^\sharp(\mu_\times)$. We have:

$$
\begin{aligned}
&\mathop{\mathbb{E}}_{(w_1, w_2)\sim\eta_\times} [d] \\
&= 1 - \mathop{\Pr}_{(w_1, w_2)\sim\eta_\times}[d = 0] + \mathop{\Pr}_{(w_1, w_2)\sim\eta_\times}[d = 2] \\
&= 1 - \mathop{\Pr}_{(m_1, m_2)\sim\mu_\times}[m_1(p) = p_0 \wedge \mathcal{S}_G(m_1(w), m_1(p), m_1(v))] \\
&\quad + \mathop{\Pr}_{(m_1, m_2)\sim\mu_\times}[m_1(p) \neq p_0 \wedge (\mathcal{S}_G(m_1(w), m_1(p), m_1(v)) \neq \mathcal{S}_G(m_2(w), m_2(p), m_2(v)))] \\
&\leq 1 - \mathop{\Pr}_{(m_1, m_2)\sim\mu_\times}[m_1(p) = p_0 \wedge \mathcal{S}_G(m_1(w), m_1(p), m_1(v))] \\
&\quad + \mathop{\Pr}_{(m_1, m_2)\sim\mu_\times}[m_1(p) \neq p_0 \wedge \neg(\mathcal{S}_G(m_1(w), m_1(p), m_1(v)) \wedge \mathcal{S}_G(m_2(w), m_2(p), m_2(v)))]
\end{aligned}
$$

To bound the first probability, the probability of selecting particle p_0 is $1/s$ and the selected particle is safe at v if it avoids the other $s-1$ locations and their neighbors (at most $(s-1)(D+1)$ bad locations). To bound the second probability, the probability of selecting a particle not equal to p_0 is $1 - 1/s$, and p is safe at v on both sides unless we select the position a, b, or one of their neighbors (at most $2(D+1)$ bad points). Putting everything together, we conclude

$$
\mathop{\mathbb{E}}_{(w_1, w_2)\sim\eta_\times}[d(w_1, w_2)] \leq 1 - \frac{1}{s}\left(1 - \frac{(s-1)(D+1)}{n}\right) + \left(1 - \frac{1}{s}\right)\left(\frac{2(D+1)}{n}\right)
$$

$$
= \left(1 - \frac{1}{s}\right)\left(1 + \frac{3(D+1)}{n}\right) \triangleq \beta.
$$

By the path coupling theorem (Theorem 3.6.2), we can bound the distance between the T-step distributions $\eta_1^{(T)}, \eta_2^{(T)}$ over final placements from any two initial placements:

$$
d_{\text{tv}}\left(\eta_1^{(T)}, \eta_2^{(T)}\right) \leq \beta^T \cdot s.
$$

When $\beta < 1$, the distributions converge exponentially quickly. $\qquad\square$

Remark 3.6.6. Like the Glauber dynamics, this Markov chain also has the uniform distribution over safe placements as a stationary distribution. Theorem 3.6.5 shows the distribution over placements converges exponentially quickly to this distribution when $\beta < 1$, starting from any safe placement.

Bubley and Dyer (1997) actually proved a stronger version of Theorem 3.6.5:

$$
\mathop{\mathbb{E}}_{(w_1, w_2)\sim\eta_\times}[d(w_1, w_2)] \leq \left(1 - \frac{1}{s}\right)\left(1 + \frac{2(D+1)}{n}\right),
$$

which is sharper than our bound

$$\mathop{\mathbb{E}}_{(w_1,w_2)\sim\eta_\times}[d(w_1,w_2)] \le \left(1-\frac{1}{s}\right)\left(1+\frac{3(D+1)}{n}\right).$$

Their analysis used the maximal coupling to couple the state distributions from sampling and updating the placement, giving a tighter bound on the expected distance.

While it is technically possible to extend ×PRHL with a sampling rule modeling the maximal coupling, with the corresponding product program drawing correlated samples from the witness distribution, the result would be somewhat unnatural. First, we would need to describe the witness distribution precisely—the maximal coupling μ of two distributions μ_1,μ_2 satisfies the equation

$$d_{\mathrm{tv}}(\mu_1,\mu_2) = \mathop{\Pr}_{(a_1,a_2)\sim\mu}[a_1 \ne a_2]$$

but the probabilities of other events are not specified. In general, there could be multiple possible witnesses to the maximal coupling, and it is unclear which witness should the canonical choice.

Furthermore, the maximal coupling is defined in terms of the probability of samples being different. This makes the maximal coupling a poor fit for our logics, which describe the support of the witness via probabilistic lifting. We would only be able to prove the trivial post-condition after applying the maximal coupling; the properties of the maximal coupling would then enter as axioms when verifying the coupled product.

3.7 Comparison with existing product programs

Product constructions reduce a relational property of two programs to a non-relational property of a single program, so that more standard techniques can be brought to bear. We close this chapter by comparing our coupled product to other existing constructions.

Almost all product constructions were originally designed with non-probabilistic programs in mind, targeting relational properties like information flow and correctness of compiler transformations. These approaches include *self composition* (Barthe, D'Argenio, and Rezk, 2011b), the *cross product* (Zaks and Pnueli, 2008), *type-directed* product programs (Terauchi and Aiken, 2005), and more (Barthe, Crespo, and Kunz, 2011a, 2013a). A basic consideration is how to handle different control flow in the two programs. If the two programs have the same shape and always take the same branches, the product program can interleave instructions from the two programs. If the two programs are very different or if the control flows are not synchronized, an asynchronous construction can combine the two programs sequentially.

These approaches have different strengths and weaknesses. By placing corresponding instructions close to one another, synchronized constructions can better leverage similarity between programs and can often be verified with simpler invariants and more local reasoning. However, asynchronous products apply to a wider range of programs. The design of ×PRHL, and in particular the asynchronous rule [WHILE-GEN], allows product programs that are both synchronous and asynchronous.

Probabilistic programs introduce additional challenges for product constructions. Existing constructions can be blindly applied to randomized programs, but the results use two independent sources of randomness, and are difficult to reason about—there is no coordination between the two programs on sampling instructions, whether the construction has a synchronous structure or not. A notable exception is the product construction by Barthe, Gaboardi, Gallego Arias, Hsu, Kunz, and Strub (2014b), which is specialized to proving differential privacy. Their construction eliminates the random sampling statements entirely, yielding a synchronized, non-probabilistic product. In fact, their product is based on a variant of probabilistic couplings called *approximate liftings*; we turn to these couplings in the rest of the thesis.

Chapter 4

Approximate couplings for differential privacy

The first half of this thesis connected proofs by coupling with the logic PRHL, using ideas from the former to enhance the latter. We now explore a similar connection in reverse, using concepts from program logics to develop a novel form of probabilistic coupling and a new proof technique. Our starting point is APRHL, an *approximate* version of PRHL proposed by Barthe et al. (2013c) for verifying *differential privacy*, a statistical notion of data privacy. This logic was originally based on an approximate version of probabilistic lifting. By interpreting approximate liftings as a generalization of probabilistic coupling and reverse-engineering an approximate version of proof by coupling from APRHL, we can give a powerful method to prove differential privacy.

After briefly reviewing differential privacy (Section 4.1), we propose a new definition of approximate lifting and explore its theoretical properties (Section 4.2); our approximate liftings are a natural, approximate version of probabilistic couplings. To build approximate couplings, we review a core version APRHL (Section 4.3) and extract a proof technique inspired by the logic, called *proof by approximate coupling* (Section 4.4). We then extend APRHL with proof rules modeling new approximate couplings (Section 4.5) and a principle called *pointwise equality* for proving differential privacy (Section 4.6). As applications, we give new proofs of privacy for the *Report-noisy-max* and *Sparse Vector* mechanisms (Section 4.7). Our approximate coupling proofs are significantly cleaner than existing arguments, and can be formalized in APRHL, enabling the first formal privacy proofs for these mechanisms. Finally, we survey other verification techniques for differential privacy, and research on approximate liftings (Section 4.8).

4.1 Differential privacy preliminaries

Differential privacy, proposed by Dwork, McSherry, Nissim, and Smith (2006), is a strong, probabilistic notion of data privacy that has attracted intensive attention across computer science and beyond. Differential privacy is a relational property of probabilistic programs.

Definition 4.1.1. Let ε, δ be non-negative parameters. Consider a set $\mathcal{D}$ with a binary *adjacency* relation *Adj*; we sometimes call $\mathcal{D}$ the set of *databases*. Let the *range* $\mathcal{R}$ be a set of possible outputs. A function $M : \mathcal{D} \to \mathbf{Distr}(\mathcal{R})$—often called a *mechanism*—is (ε, δ)-*differentially private* if for all pairs of adjacent inputs $(d, d') \in Adj$ and all subsets $\mathcal{S} \subseteq \mathcal{R}$ of outputs, we have

$$M(d)(\mathcal{S}) \leq \exp(\varepsilon) \cdot M(d')(\mathcal{S}) + \delta.$$

When $\delta = 0$, we say M is ε-*differentially private*.

The adjacency relation describes which pairs of databases should lead to approximately indistinguishable outputs—intuitively, which pairs of databases differ only in the data of a single person. For instance, if a database is a set of records belonging to different people, we can consider two databases to be adjacent if they are identical except for an additional individual's record in one database. Then under differential privacy, a mechanism's output

44

must be nearly the same whether any single individual's private data is part of the input or not. The degree of similarity—and the strength of the privacy guarantee—are governed by the parameters ε and δ: smaller values give stronger guarantees, while larger values give weaker guarantees.

While typical notions of adjacency are symmetric, much of the theory of differential privacy applies to arbitrary relations. However, there are a few notable results that crucially need a symmetric adjacency relation—we will highlight these cases as they arise.

Standard private mechanisms

The most basic example of a differentially private mechanism is the *Laplace mechanism*, which evaluates a numeric query on a database and adds random noise drawn from the Laplace distribution. For instance, the target query could compute the average age, or count the number of patients with a certain disease. While the Laplace distribution is a continuous distribution over the real numbers, we work with a discrete version to avoid measure-theoretic technicalities. For concreteness we take the samples to be integers; our results can be easily adapted to finer discretizations.[1]

Definition 4.1.2. Let $\varepsilon > 0$. The *(discrete) Laplace distribution* with parameter ε, written $\mathbf{Lap}_\varepsilon$, is the distribution over the integers where $v \in \mathbb{Z}$ has probability proportional to $\exp(-|v| \cdot \varepsilon)$:

$$\mathbf{Lap}_\varepsilon(v) \triangleq \frac{\exp(-|v| \cdot \varepsilon)}{W},$$

with $W \triangleq \sum_{z \in \mathbb{Z}} \exp(-|z| \cdot \varepsilon)$. We write $\mathbf{Lap}_\varepsilon(t)$ for the Laplace distribution with mean $t \in \mathbb{Z}$; sampling from this distribution is equivalent to sampling from $\mathbf{Lap}_\varepsilon$ and adding t.

Let $q : \mathcal{D} \to \mathbb{Z}$ be an integer-valued query. The *Laplace mechanism* with parameter ε takes a database $d \in \mathcal{D}$ as input and returns a sample from $\mathbf{Lap}_\varepsilon(q(d))$. This mechanism is also known as the *ε-geometric mechanism* (Ghosh, Roughgarden, and Sundararajan, 2012).

If the query takes similar values on adjacent databases, the Laplace mechanism is differentially private. The privacy parameters depend on the *sensitivity* of the query—the more the answers may differ on adjacent databases, the weaker the privacy guarantee.

Theorem 4.1.3 (Dwork et al. (2006)). *A query $q : \mathcal{D} \to \mathbb{Z}$ is k-sensitive if $|q(d) - q(d')| \leq k$ for every pair of adjacent databases. Releasing a k-sensitive query with the Laplace mechanism with parameter ε is $(k \cdot \varepsilon, 0)$-differentially private.*

Composition theorems

Differential privacy is closed under several notions of composition, making it easy to build new private algorithms out of private components. The *sequential*, or *standard composition theorem* is the most basic example. When running two private computations in sequence—where the second computation may use the input database as well as the randomized output from the first computation—the privacy guarantee should weaken, since we run more analyses on the data. Indeed, the privacy parameters simply add up.

Theorem 4.1.4 (Dwork et al. (2006)). *Let $M : \mathcal{D} \to \mathbf{Distr}(\mathcal{R})$ be (ε, δ)-differentially private and let $M' : \mathcal{R} \times \mathcal{D} \to \mathbf{Distr}(\mathcal{R})$ be such that $M'(r, -) : \mathcal{D} \to \mathbf{Distr}(\mathcal{R})$ is (ε', δ')-differentially private for every $r \in \mathcal{R}$. Given a database $d \in \mathcal{D}$, sampling r from $M(d)$ and then returning a sample from $M'(r, d)$ is $(\varepsilon + \varepsilon', \delta + \delta')$-differentially private.*

This useful theorem has two immediate consequences. First, if M' depends only on its first argument r and ignores its database argument d, then $M'(r, -)$ is $(0, 0)$-differentially private. So, transforming the output of a differentially-private algorithm does not degrade privacy; this property is also called *closure under post-processing*.

---

[1]More precisely, any discretization closed under addition.

Second, by repeatedly applying the composition theorem, the composition of n separate (ε, δ)-differentially private mechanisms is $(n\varepsilon, n\delta)$-differentially private. In certain parameter ranges, an alternative, *advanced composition* theorem can bound the privacy level with a smaller ε at the cost of a slightly larger δ. This result crucially assumes a symmetric adjacency relation.

Theorem 4.1.5 (Dwork, Rothblum, and Vadhan (2010))**.** *Fix a* symmetric *adjacency relation on $\mathcal{D}$. Let $f_i : \mathcal{R} \times \mathcal{D} \to$* **Distr**$(\mathcal{R})$ *be a sequence of n functions such that for every $r \in \mathcal{R}$, the functions $f_i(r, -) : \mathcal{D} \to$* **Distr**$(\mathcal{R})$ *are (ε, δ)-differentially private. Then for every $\omega \in (0, 1)$, the mechanism that executes $f_1, \dots, f_n$ in sequence and returns the final output is $(\varepsilon^*, \delta^*)$-differentially private for*

$$\varepsilon^* = \varepsilon\sqrt{2n\ln(1/\omega)} + n\varepsilon(e^\varepsilon - 1) \quad and \quad \delta^* = n\delta + \omega.$$

In particular, if we have $\varepsilon' \in (0, 1)$, $\omega \in (0, 1/2)$, and

$$\varepsilon = \frac{\varepsilon'}{2\sqrt{2n\ln(1/\omega)}},$$

a short calculation[2] shows that the composition is (ε', δ^*)-differentially private.

We omit other standard composition theorems (e.g., parallel composition) as we will not need them; readers can consult the textbook by Dwork and Roth (2014) for more information.

Remark 4.1.6. The sequential composition theorem allows reasoning about differential privacy in terms of *privacy costs*. We can imagine tracking an algorithm's privacy parameters, initially $(0, 0)$. Every time the algorithm applies an (ε, δ)-private mechanism, we increment the current parameters by (ε, δ); the final parameters give the privacy level for the whole algorithm. In this way, (ε, δ) represents the *cost* of using a private subroutine.

While this observation seems to be a restatement of the composition theorems, merely a convenient accounting method, the subtlety lies in how the costs are computed. The key point is that outputs from previous private mechanisms are assumed to be *equal* when computing the cost of subsequent operations. Changing the perspective a bit, we can pay cost (ε, δ) to assume two outputs in related runs of an (ε, δ)-private mechanism are equal. We can begin to see the rough contours of a proof by coupling; we will soon make this idea more precise.

4.2   Approximate liftings

Differential privacy is closely related to an approximate version of probabilistic lifting first proposed by Barthe et al. (2013c) and refined in later work (Barthe and Olmedo, 2013; Olmedo, 2014). These liftings are defined in terms of a distance on distributions.

---

[2] Note $e^\varepsilon - 1 \le 2\varepsilon$ for $\varepsilon \in (0, 1)$ by convexity of $e^\varepsilon - 2\varepsilon - 1$. Then

$$\sqrt{2n\ln(1/\omega)}\varepsilon + n\varepsilon(e^\varepsilon - 1) \le \sqrt{2n\ln(1/\omega)}\varepsilon + 2n\varepsilon^2$$

$$= \frac{\varepsilon'}{2} + \frac{\varepsilon'}{2} \cdot \frac{\varepsilon'}{2\ln(1/\omega)}$$

$$\le \frac{\varepsilon'}{2} + \frac{\varepsilon'}{2} = \varepsilon',$$

where the last inequality is because $\omega \in (0, 1/2)$ and $\varepsilon' \in (0, 1)$, and the last factor is maximized at $\varepsilon' = 1$ and $\omega = 1/2$:

$$\frac{\varepsilon'}{2\ln(1/\omega)} \le \frac{1}{2\ln(2)} < 1.$$

Definition 4.2.1. Let μ_1, μ_2 be sub-distributions over $\mathcal{A}$. The ε-*distance* is defined as

$$d_\varepsilon(\mu_1, \mu_2) \triangleq \max_{\mathcal{S} \subseteq \mathcal{A}}(\mu_1(\mathcal{S}) - \exp(\varepsilon) \cdot \mu_2(\mathcal{S})).$$

This quantity is non-negative since the right-hand side is zero for the empty subset $\mathcal{S} = \varnothing$, but it is not a proper metric—it is not symmetric and the triangle inequality does not hold.[3] If $M : \mathcal{D} \to \mathbf{Distr}(\mathcal{R})$ is a mechanism with $d_\varepsilon(M(d_1), M(d_2)) \leq \delta$ for every pair of adjacent d_1, d_2, then M is (ε, δ)-differentially private.

We are now ready to define approximate liftings.

Definition 4.2.2. Let μ_1, μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$ respectively and let $\mathcal{R} \subseteq \mathcal{A}_1 \times \mathcal{A}_2$ be a relation. Let $\star$ be a distinguished element disjoint from $\mathcal{A}_1$ and $\mathcal{A}_2$; we write $\mathcal{S}^\star$ for the set $\mathcal{S} \cup \{\star\}$, and $\mathcal{R}^\star$ for the relation $\mathcal{R} \cup (\mathcal{A}_1 \times \{\star\}) \cup (\{\star\} \times \mathcal{A}_2)$ on $\mathcal{A}_1^\star \times \mathcal{A}_2^\star$. Two sub-distributions μ_L, μ_R over $\mathcal{A}_1^\star \times \mathcal{A}_2^\star$ are said to be *witnesses* for the (ε, δ)-*approximate $\mathcal{R}$-lifting* of (μ_1, μ_2) if:

1. $\pi_1(\mu_L) = \mu_1$ and $\pi_2(\mu_R) = \mu_2$;

2. $\mathrm{supp}(\mu_L) \cup \mathrm{supp}(\mu_R) \subseteq \mathcal{R}^\star$; and

3. $d_\varepsilon(\mu_L, \mu_R) \leq \delta$.

In the first point μ_1 and μ_2 are implicitly interpreted as distributions over $\mathcal{A}_1^\star$ and $\mathcal{A}_2^\star$ (i.e., placing zero probability on $\star$). We call these conditions the marginal, support, and distance conditions, respectively.

The sub-distributions μ_L and μ_R are called *left* and *right witnesses* of the lifting. When the particular witnesses are not important, μ_1 and μ_2 are said to be related by the (ε, δ)-*lifting of $\mathcal{R}$*, denoted

$$\mu_1 \, \mathcal{R}^{\sharp(\varepsilon, \delta)} \, \mu_2.$$

Our definition generalizes an earlier definition of approximate lifting by Barthe and Olmedo (2013). The chief novelty is the element $\star$, which ensures each element in $\mathcal{A}_1$ and $\mathcal{A}_2$ can be related to some element under $\mathcal{R}$ (namely, $\star$). Somewhat paradoxically, the larger space of witnesses lets us assume more structure on the witness distributions without loss of generality, making it easier to manipulate and construct approximate liftings.

Useful consequences

The existence of an approximate lifting between two distributions can imply useful properties about the two distributions. Many of these consequences recall properties from Section 2.1, with quantitative corrections for the parameters (ε, δ).

Proposition 4.2.3. *Let $M : \mathcal{D} \to \mathbf{Distr}(\mathcal{R})$ be a randomized algorithm. If for every pair of adjacent inputs (d_1, d_2) the output distributions are related by an approximate lifting*

$$M(d_1) \, (=)^{\sharp(\varepsilon, \delta)} \, M(d_2),$$

then M is (ε, δ)-differentially private.

Proof. Fix a pair of adjacent inputs (d_1, d_2), and let μ_L, μ_R be the witnesses to the approximate lifting of the output distributions. For any subset $\mathcal{S} \subseteq \mathcal{R}$ of outputs, we have

$$\begin{aligned} M(d_1)(\mathcal{S}) &= \mu_L(\mathcal{S} \times \mathcal{R}^\star) && \text{(first marginal)} \\ &= \mu_L(\{(s, s) \mid s \in \mathcal{S}\} \cup \mathcal{S} \times \{\star\}) && \text{(support)} \end{aligned}$$

---

[3]Technically, ε-distance is an f-divergence with $f(t) = \max(t - \exp(\varepsilon), 0)$

$$\leq \exp(\varepsilon) \cdot \mu_R(\{(s,s) \mid s \in \mathcal{S}\} \cup \mathcal{S} \times \{\star\}) + \delta \qquad \text{(distance)}$$
$$\leq \exp(\varepsilon) \cdot \mu_R(\mathcal{R}^\star \times \mathcal{S}^\star) + \delta \qquad \text{(support)}$$
$$= \exp(\varepsilon) \cdot M(d_2)(\mathcal{S}) + \delta. \qquad \text{(second marginal)}$$

Thus M is (ε, δ)-differentially private.   □

The approximate lifted version of implication is also useful.

Proposition 4.2.4. *Let μ_1, μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$, and consider subsets $\mathcal{S}_1 \subseteq \mathcal{A}_1$, $\mathcal{S}_2 \subseteq \mathcal{A}_2$. Suppose we have an approximate lifting*

$$\mu_1 \ \{(a_1, a_2) \mid a_1 \in \mathcal{S}_1 \to a_2 \in \mathcal{S}_2\}^{\sharp(\varepsilon, \delta)} \ \mu_2.$$

Then $\mu_1(\mathcal{S}_1) \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{S}_2) + \delta$.

Proof. Let μ_L, μ_R witness the approximate lifting. Then,

$$\mu_1(\mathcal{S}_1) = \mu_L(\mathcal{S}_1 \times \mathcal{R}^\star) \qquad \text{(first marginal)}$$
$$= \mu_L(\mathcal{S}_1 \times \mathcal{S}_2 \cup \mathcal{S}_1 \times \{\star\}) \qquad \text{(support)}$$
$$\leq \exp(\varepsilon) \cdot \mu_R(\mathcal{S}_1 \times \mathcal{S}_2 \cup \mathcal{S}_1 \times \{\star\}) + \delta \qquad \text{(distance)}$$
$$\leq \exp(\varepsilon) \cdot \mu_R(\mathcal{R}^\star \times \mathcal{S}_2^\star) + \delta \qquad \text{(support)}$$
$$= \exp(\varepsilon) \cdot \mu_2(\mathcal{S}_2) + \delta \qquad \text{(second marginal)}$$

as desired.   □

We will see a partial converse in the next chapter (Theorem 5.3.1).

Structural properties

Approximate liftings satisfy several natural structural properties. First of all, they generalize exact liftings.

Proposition 4.2.5. *Let μ_1, μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$ with equal weights. We have the equivalence*

$$\mu_1 \ \mathcal{R}^\sharp \ \mu_2 \quad \text{if and only if} \quad \mu_1 \ \mathcal{R}^{\sharp(0,0)} \ \mu_2.$$

Proof. The forward direction follows by taking both witnesses of the approximate lifting to be the witness of the exact lifting. For the reverse direction, let μ_L, μ_R witness the approximate lifting. We have $d_0(\mu_L, \mu_R) \leq 0$ so $\mu_L(a_1, a_2) \leq \mu_R(a_1, a_2)$ for every pair $(a_1, a_2) \in \mathcal{A}_1 \times \mathcal{A}_2$. Since μ_1 and μ_2 have equal weights, the marginal conditions imply $|\mu_L| = |\mu_R|$ and hence $\mu_L = \mu_R$. Since $\mu_L(\{\star\} \times \mathcal{A}_2) = \mu_R(\mathcal{A}_1 \times \{\star\}) = 0$, restricting to $\mathcal{A}_1 \times \mathcal{A}_2$ gives a witness for the exact lifting as desired.   □

Second, we may assume witnesses only use pairs in the product of the supports of the two related distributions.

Proposition 4.2.6. *Let μ_1 and μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$ with an approximate lifting*

$$\mu_1 \ \mathcal{R}^{\sharp(\varepsilon, \delta)} \ \mu_2.$$

Then there are witnesses with support contained in $supp(\mu_1)^\star \times supp(\mu_2)^\star$.

This property is natural—μ_1 and μ_2 are fully defined by their probabilities on supporting elements, so the witnesses shouldn't need to use other elements. However, witnesses to an approximate lifting may have positive mass on points $(a_1, a_2) \notin supp(\mu_1) \times supp(\mu_2)$ since the marginal conditions only constrain one marginal of μ_L and μ_R; mass can be distributed arbitrarily along the unconstrained component. In fact, this support property does not hold for prior definitions of approximate lifting. In our definition, the $\star$ element serves as a canonical element where mass outside of $supp(\mu_1) \times supp(\mu_2)$ can be located.

Proof. Let μ_L and μ_R witness the approximate lifting and let $\mathcal{S}_i \triangleq \mathrm{supp}(\mu_i)$ for $i \in \{1, 2\}$. We construct witnesses η_L, η_R by shifting mass on points outside the support to $\star$, while preserving the marginals:

$$\eta_L(a_1, a_2) \triangleq \begin{cases} \mu_L(a_1, a_2) & : (a_1, a_2) \in \mathcal{S}_1 \times \mathcal{S}_2 \\ \sum_{a_2' \in \mathcal{A}_2^\star \setminus \mathcal{S}_2} \mu_L(a_1, a_2') & : a_2 = \star \\ 0 & : \text{otherwise} \end{cases}$$

$$\eta_R(a_1, a_2) \triangleq \begin{cases} \mu_R(a_1, a_2) & : (a_1, a_2) \in \mathcal{S}_1 \times \mathcal{S}_2 \\ \sum_{a_1' \in \mathcal{A}_1^\star \setminus \mathcal{S}_1} \mu_R(a_1', a_2) & : a_1 = \star \\ 0 & : \text{otherwise} \end{cases}$$

It is straightforward to check $\pi_1(\eta_L) = \pi_1(\mu_L) = \mu_1$ and $\pi_2(\eta_R) = \pi_2(\mu_R) = \mu_2$, and η_L, η_R have the necessary supports. It only remains to check the distance condition. By the distance condition on μ_L and μ_R, there are non-negative constants $\delta(a_1, a_2)$ such that

$$\mu_L(a_1, a_2) \leq \exp(\varepsilon) \cdot \mu_R(a_1, a_2) + \delta(a_1, a_2)$$

for each $(a_1, a_2) \in \mathcal{A}_1^\star \times \mathcal{A}_2^\star$, with sum at most δ. We define new constants

$$\delta'(a_1, a_2) \triangleq \begin{cases} \delta(a_1, a_2) & : (a_1, a_2) \in \mathcal{S}_1 \times \mathcal{S}_2 \\ \sum_{a_2' \in \mathcal{A}_2^\star \setminus \mathcal{S}_2} \delta(a_1, a_2') & : a_2 = \star \\ 0 & : \text{otherwise} \end{cases}$$

and we claim

$$\eta_L(a_1, a_2) \leq \exp(\varepsilon) \cdot \eta_R(a_1, a_2) + \delta'(a_1, a_2).$$

This is clear on $\mathcal{S}_1 \times \mathcal{S}_2$ and also when $a_1 = \star$, since $\eta_L(\star, a_2) = 0$. When $a_2 = \star$, unfolding definitions gives

$$\begin{aligned} \eta_L(a_1, \star) &= \sum_{a_2' \in \mathcal{A}_2^\star \setminus \mathcal{S}_2} \mu_L(a_1, a_2') \\ &\leq \sum_{a_2' \in \mathcal{A}_2^\star \setminus \mathcal{S}_2} \exp(\varepsilon) \cdot \mu_R(a_1, a_2') + \delta(a_1, a_2') \\ &= \sum_{a_2' \in \mathcal{A}_2^\star \setminus \mathcal{S}_2} \delta(a_1, a_2') \\ &= \exp(\varepsilon) \cdot \eta_R(a_1, \star) + \delta'(a_1, \star) \end{aligned}$$

where the penultimate equality is because $\mu_R(a_1, a_2') = 0$ for $a_2' \notin \mathcal{S}_2$, and the last equality is because $\eta_R(a_1, \star) = 0$ by definition. Finally,

$$\sum_{(a_1, a_2) \in \mathcal{A}_1^\star \times \mathcal{A}_2^\star} \delta'(a_1, a_2) = \sum_{(a_1, a_2) \in \mathcal{A}_1 \times \mathcal{A}_2^\star} \delta(a_1, a_2) \leq \delta$$

so the distance condition $d_\varepsilon(\eta_L, \eta_R) \leq \delta$ holds. Thus η_L and η_R witness the approximate lifting. $\square$

Approximate liftings are also stable under mappings.

Theorem 4.2.7. *Let μ_1 and μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$. If we have functions $f_i : \mathcal{A}_i \to \mathcal{B}_i$ for $i \in \{1, 2\}$, and a relation $\mathcal{R} \subseteq \mathcal{B}_1 \times \mathcal{B}_2$, then*

$$\mu_1 \{(a_1, a_2) \in \mathcal{A}_1 \times \mathcal{A}_2 \mid f_1(a_1) \; \mathcal{R} \; f_2(a_2)\}^{\sharp(\varepsilon, \delta)} \mu_2$$

if and only if

$$f_1^\sharp(\mu_1) \, \{(b_1, b_2) \in \mathcal{B}_1 \times \mathcal{B}_2 \mid b_1 \, \mathcal{R} \, b_2\}^{\sharp(\varepsilon, \delta)} \, f_2^\sharp(\mu_2).$$

(Recall $f : \mathcal{A} \to \mathcal{B}$ can be lifted to a map $f^\sharp : \mathbf{SDistr}(\mathcal{A}) \to \mathbf{SDistr}(\mathcal{B})$ on sub-distributions.)

This theorem roughly says that we can change the basis of an approximate lifting; namely, the ground sets of μ_1 and μ_2 and the ambient space of the relation. Several useful consequences follow. First, if we take f_1 and f_2 to inject from $\mathrm{supp}(\mu_1)$ and $\mathrm{supp}(\mu_2)$ into $\mathcal{B}_1$ and $\mathcal{B}_2$, the reverse direction recovers Proposition 4.2.6. Second, if $\mathcal{E}$ is a set of equivalence classes of $\mathcal{A}$ and $\mu/\mathcal{E} \in \mathbf{SDistr}(\mathcal{E})$ is the induced distribution over equivalence classes, taking $f_1, f_2 : \mathcal{A} \to \mathcal{E}$ to map an element to its equivalence class and $\mathcal{R}$ to be the equivalence relation $=_{\mathcal{E}}$ recovers a result by Barthe and Olmedo (2013, Proposition 8):

$$\mu_1 \, (=_{\mathcal{E}})^{\sharp(\varepsilon, \delta)} \, \mu_2 \iff \mu_1/\mathcal{E} \, (=)^{\sharp(\varepsilon, \delta)} \, \mu_2/\mathcal{E}.$$

We frequently apply Theorem 4.2.7 with f_1 and f_2 projecting a memory to the value in variables x_1 and x_2; by the reverse direction, we can extend a lifting of the distributions over x_1 and x_2 to a lifting of distributions over whole memories.

Proof. For the forward direction, take witnesses $\mu_L, \mu_R \in \mathbf{SDistr}(\mathcal{A}_1^\star \times \mathcal{A}_2^\star)$ and define witnesses for the desired approximate lifting $\eta_L \triangleq (f_1^\star \times f_2^\star)^\sharp(\mu_L)$ and $\eta_R \triangleq (f_1^\star \times f_2^\star)^\sharp(\mu_R)$, where $f_1^\star \times f_2^\star$ maps $(a_1, a_2) \mapsto (f_1(a_1), f_2(a_2))$ and maps $\star$ to $\star$ in both components. The support condition is clear, the marginal requirement is clear, and the distance requirement follows easily: for any set $\mathcal{S} \subseteq \mathcal{B}_1^\star \times \mathcal{B}_2^\star$, apply the distance condition on μ_L, μ_R for the set $(f_1^\star \times f_2^\star)^{-1}(\mathcal{S})$.

For the reverse direction, let $\eta_L, \eta_R \in \mathbf{SDistr}(\mathcal{B}_1^\star \times \mathcal{B}_2^\star)$ witness the second approximate lifting. By Proposition 4.2.6, without loss of generality $\mathrm{supp}(\eta_L)$ and $\mathrm{supp}(\eta_R)$ are contained in

$$\mathrm{supp}(f_1^\sharp(\mu_1))^\star \times \mathrm{supp}(f_2^\sharp(\mu_2))^\star \subseteq f_1(\mathcal{A}_1)^\star \times f_2(\mathcal{A}_2)^\star. \tag{4.1}$$

We construct a pair of witnesses $\mu_L, \mu_R \in \mathbf{SDistr}(\mathcal{A}_1^\star \times \mathcal{A}_2^\star)$ to the first approximate lifting. The basic idea is to define μ_L and μ_R based on equivalence classes of elements in $\mathcal{A}_i$ mapping to each $b_i \in \mathcal{B}_i$, smoothing out the probabilities within each class to guarantee the distance condition. To begin, for $a_i \in \mathcal{A}_i$ and $i \in \{1, 2\}$ we define

$$[a_i]_{f_i} \triangleq f_i^{-1}(f_i(a_i)) \quad \text{and} \quad \alpha_i(a_i) \triangleq \frac{\mu_i(a_i)}{\mu_i([a_i]_{f_i})}.$$

We take $\alpha_i(a_i) = 0$ when $\mu_i([a_i]_{f_i}) = 0$, and we let $\alpha_i(\star) = 0$. We define μ_L and μ_R as

$$\mu_L(a_1, a_2) \triangleq \alpha_L(a_1, a_2) \cdot \eta_L(f_1^\star(a_1), f_2^\star(a_2))$$
$$\mu_R(a_1, a_2) \triangleq \alpha_R(a_1, a_2) \cdot \eta_R(f_1^\star(a_1), f_2^\star(a_2)),$$

where

$$\alpha_L(a_1, a_2) \triangleq \begin{cases} \alpha_1(a_1) \cdot \alpha_2(a_2) & : a_2 \neq \star \\ \alpha_1(a_1) & : a_2 = \star \end{cases} \quad \text{and} \quad \alpha_R(a_1, a_2) \triangleq \begin{cases} \alpha_1(a_1) \cdot \alpha_2(a_2) & : a_1 \neq \star \\ \alpha_2(a_2) & : a_1 = \star. \end{cases}$$

The support and marginal conditions follow from the corresponding properties of η_L, η_R, e.g.,

$$\pi_1(\mu_L)(a_1) = \sum_{a_2 \in \mathcal{A}_2^\star} \alpha_L(a_1, a_2) \cdot \eta_L(f_1^\star(a_1), f_2^\star(a_2))$$

$$= \alpha_1(a_1) \cdot \eta_L(f_1^\star(a_1), \star) + \sum_{a_2 \in \mathcal{A}_2} \alpha_1(a_1) \cdot \alpha_2(a_2) \cdot \eta_L(f_1(a_1), f_2(a_2))$$

$$\begin{aligned}
&= \alpha_1(a_1) \left(\eta_L(f_1^\star(a_1), \star) + \sum_{b_2 \in f_2(\mathcal{A}_2)} \eta_L(f_1^\star(a_1), b_2) \sum_{a_2 \in f_2^{-1}(b_2)} \alpha_2(a_2) \right) \\
&= \alpha_1(a_1) \left(\eta_L(f_1^\star(a_1), \star) + \sum_{b_2 \in f_2(\mathcal{A}_2)} \eta_L(f_1(a_1), b_2) \right) \\
&= \alpha_1(a_1) \sum_{b_2 \in \mathcal{B}_2^\star} \eta_L(f_1^\star(a_1), b_2) = \alpha_1(a_1) \cdot \mu_1([a_1]_{f_1}) = \mu_1(a_1).
\end{aligned}$$

The last equality replaces the sum over $b_2 \in f_2(\mathcal{A}_2)^\star$ with a sum over $b_2 \in \mathcal{B}_2^\star$; this holds since the support of $f_2^\sharp(\mu_2)$ is contained in $f_2(\mathcal{A}_2)$ so we may assume $\eta_L(f_1(a_1), b_2) = 0$ for all b_2 outside of $f_2(\mathcal{A}_2)^\star$ by Eq. (4.1). Then we conclude by the marginal condition $\pi_1(\eta_L) = f_1^\sharp(\mu_1)$. The second marginal is similar.

To check the distance condition $d_\varepsilon(\mu_L, \mu_R) \le \delta$, since $d_\varepsilon(\eta_L, \eta_R) \le \delta$ there exists non-negative $\delta(b_1, b_2)$ with

$$\eta_L(b_1, b_2) \le \exp(\varepsilon) \cdot \eta_R(b_1, b_2) + \delta(b_1, b_2)$$

and $\sum_{b_1, b_2} \delta(b_1, b_2) \le \delta$. We may take $\delta(\star, b_2) = 0$ for all $b_2 \in \mathcal{B}_2$, since $\eta_L(\star, b_2) = 0$ by the marginal condition. We claim that for any $(a_1, a_2) \in \mathcal{A}_1^\star \times \mathcal{A}_2^\star$, we have $\mu_L(a_1, a_2) \le \exp(\varepsilon) \cdot \mu_R(a_1, a_2) + \zeta(a_1, a_2)$ where

$$\zeta(a_1, a_2) \triangleq \alpha_L(a_1, a_2) \cdot \delta(f_1^\star(a_1), f_2^\star(a_2)).$$

Let $b_i \triangleq f_i^\star(a_i)$ and consider $(a_1, a_2) \in \mathcal{A}_1^\star \times \mathcal{A}_2^\star$. If $a_1 = \star$ we can immediately bound

$$\mu_L(\star, a_2) = 0 \le \exp(\varepsilon) \cdot \mu_R(\star, a_2) + \zeta(\star, a_2).$$

Otherwise $a_1 \ne \star$ and we can bound

$$\begin{aligned}
\mu_L(a_1, a_2) &= \alpha_L(a_1, a_2) \cdot \eta_L(f_1^\star(a_1), f_2^\star(a_2)) \\
&\le \alpha_L(a_1, a_2) \cdot (\exp(\varepsilon) \cdot \eta_R(f_1^\star(a_1), f_2^\star(a_2)) + \delta(f_1^\star(a_1), f_2^\star(a_2))) \\
&= \exp(\varepsilon) \cdot (\alpha_R(a_1, a_2) \cdot \eta_R(f_1^\star(a_1), f_2^\star(a_2))) + \alpha_L(a_1, a_2) \cdot \delta(f_1^\star(a_1), f_2^\star(a_2)) \\
&= \exp(\varepsilon) \cdot \mu_R(a_1, a_2) + \alpha_L(a_1, a_2) \cdot \delta(f_1^\star(a_1), f_2^\star(a_2)) \\
&= \exp(\varepsilon) \cdot \mu_R(a_1, a_2) + \zeta(a_1, a_2).
\end{aligned}$$

The third line changes from α_L to α_R in the first term since $\alpha_L(a_1, a_2) \ne \alpha_R(a_1, a_2)$ exactly when $a_2 = \star$, when $\eta_R(f_1^\star(a_1), f_2^\star(a_2)) = \eta_R(f_1^\star(a_1), \star) = 0$ as well.

Now we just need to bound the sum of $\zeta(a_1, a_2)$ to conclude the distance bound between η_L and η_R. First, the sum of α_L within any equivalence class is 1: for any $(b_1, b_2) \in \mathcal{B}_1 \times \mathcal{B}_2$, we have

$$\sum_{a_1 \in f_1^{-1}(b_1)} \sum_{a_2 \in f_2^{-1}(b_2)} \alpha_L(a_1, a_2) = \left(\sum_{a_1 \in f_1^{-1}(b_1)} \alpha_1(a_1) \right) \left(\sum_{a_2 \in f_2^{-1}(b_2)} \alpha_2(a_2) \right) = 1$$

by definition. Therefore,

$$\begin{aligned}
\sum_{(a_1, a_2) \in \mathcal{A}_1^\star \times \mathcal{A}_2^\star} \zeta(a_1, a_2) &= \sum_{(b_1, b_2) \in \mathcal{B}_1^\star \times \mathcal{B}_2^\star} \delta(b_1, b_2) \sum_{a_1 \in f_1^{-1}(b_1)} \sum_{a_2 \in f_2^{-1}(b_2)} \alpha_L(a_1, a_2) \\
&= \sum_{(b_1, b_2) \in \mathcal{B}_1 \times \mathcal{B}_2} \delta(b_1, b_2) + \sum_{b_1 \in \mathcal{B}_1} \delta(b_1, \star) \sum_{a_1 \in f_1^{-1}(b_1)} \alpha_1(a_1)
\end{aligned}$$

$$= \sum_{(b_1, b_2) \in \mathcal{B}_1 \times \mathcal{B}_2} \delta(b_1, b_2) + \sum_{b_1 \in \mathcal{B}_1} \delta(b_1, \star)$$

$$= \sum_{(b_1, b_2) \in \mathcal{B}_1^\star \times \mathcal{B}_2^\star} \delta(b_1, b_2) \leq \delta.$$

So for any $\mathcal{S} \subseteq \mathcal{A}_1^\star \times \mathcal{A}_2^\star$ we have $\mu_L(\mathcal{S}) \leq \exp(\varepsilon) \cdot \mu_R(\mathcal{S}) + \delta$, showing $d_\varepsilon(\mu_L, \mu_R) \leq \delta$ as desired. □

From approximate liftings to approximate couplings

Approximate liftings generalize probabilistic liftings (Proposition 4.2.5) while retaining many features of their exact counterparts: the existence of an approximate lifting with a certain support implies target properties about the two related distributions (Propositions 4.2.3 and 4.2.4), and the structural properties we saw for approximate liftings (Proposition 4.2.6 and Theorem 4.2.7) also hold for probabilistic liftings. Accordingly, we can think of approximate liftings as an approximate generalization of probabilistic coupling; we will use the term *approximate coupling* to emphasize this point of view.

Unlike probabilistic coupling, whose definition and key properties have been refined through decades of research, the proper definition of approximate coupling is not settled. Other definitions have been proposed, and the relation between the various notions is somewhat hazy. (See Section 5.6 for a more detailed comparison.) Nevertheless, we present evidence that our approximate lifting is the natural approximate counterpart of probabilistic coupling—or at least, a highly promising candidate—by showing many desirable properties hold and by exhibiting clean constructions.

However, so far we are still missing a major piece of the puzzle: how do we construct approximate couplings? In other words, what is the approximate analogue of proof by coupling? To work out what such a proof technique might look like, we take inspiration from an existing program logic for approximate liftings.

4.3   The program logic APRHL

Barthe et al. (2013c) proposed the relational program logic APRHL as an approximate version of PRHL, targeting differential privacy. The basic idea is to use approximate liftings in place of exact liftings, tracking the parameters (ε, δ) in the judgments. We briefly review the language, the judgments, and the logical rules.

The language

The language of APRHL is almost identical to the probabilistic imperative language we used for PRHL. The only difference is instead of the uniform distribution, we take the Laplace distribution as primitive:

$$\mathcal{D}\mathcal{E} := \mathbf{Lap}_\varepsilon(e).$$

The parameter ε quantifies the spread of the distribution, while the parameter e represents its mean; we treat ε as a logical variable. Similar to how we defined the Laplace mechanism (Definition 4.1.2), we interpret $\mathbf{Lap}_\varepsilon(e)$ as a discrete distribution over the integers $z \in \mathbb{Z}$:

$$(\llbracket \mathbf{Lap}_\varepsilon(e) \rrbracket_\rho m)(z) \triangleq \frac{\exp(-\llbracket \varepsilon \rrbracket_\rho \cdot |z - \llbracket e \rrbracket_\rho m|)}{W}$$

where $\llbracket e \rrbracket_\rho m$ is an integer and W normalizes the distribution to have weight 1:

$$W \triangleq \sum_{z \in \mathbb{Z}} \exp(-\llbracket \varepsilon \rrbracket_\rho \cdot |z - \llbracket e \rrbracket_\rho m|).$$

For example, the Laplace mechanism for a query $q : \mathcal{D} \to \mathbb{Z}$ can be implemented by sampling:

$$x \xleftarrow{\$} \mathbf{Lap}_\varepsilon(q(d)).$$

$$\text{SKIP} \; \frac{}{\vdash \textbf{skip} \sim_{(0,0)} \textbf{skip} : \Phi \Longrightarrow \Phi}$$

$$\text{ASSN} \; \frac{}{\vdash x_1 \leftarrow e_1 \sim_{(0,0)} x_2 \leftarrow e_2 : \Psi \{e_1\langle 1\rangle, e_2\langle 2\rangle / x_1\langle 1\rangle, x_2\langle 2\rangle\} \Longrightarrow \Psi}$$

$$\text{LAP} \; \frac{x_1, x_2 \notin \text{FV}(e_1, e_2) \qquad \Phi \triangleq |e_1\langle 1\rangle - e_2\langle 2\rangle| \le k \wedge \forall v \in \mathbb{Z}, \; \Psi \{v, v / x_1\langle 1\rangle, x_2\langle 2\rangle\}}{\vdash x_1 \xleftarrow{\$} \textbf{Lap}_\varepsilon(e_1) \sim_{(k\varepsilon, 0)} x_2 \xleftarrow{\$} \textbf{Lap}_\varepsilon(e_2) : \Phi \Longrightarrow \Psi}$$

$$\text{SEQ} \; \frac{\vdash c_1 \sim_{(\varepsilon, \delta)} c_2 : \Phi \Longrightarrow \Psi \qquad \vdash c_1' \sim_{(\varepsilon', \delta')} c_2' : \Psi \Longrightarrow \Theta}{\vdash c_1; c_1' \sim_{(\varepsilon + \varepsilon', \delta + \delta')} c_2; c_2' : \Phi \Longrightarrow \Theta}$$

$$\text{COND} \; \frac{\models \Phi \rightarrow e_1\langle 1\rangle = e_2\langle 2\rangle \qquad \vdash c_1 \sim_{(\varepsilon, \delta)} c_2 : \Phi \wedge e_1\langle 1\rangle \Longrightarrow \Psi \qquad \vdash c_1' \sim_{(\varepsilon, \delta)} c_2' : \Phi \wedge \neg e_1\langle 1\rangle \Longrightarrow \Psi}{\vdash \textbf{if } e_1 \textbf{ then } c_1 \textbf{ else } c_1' \sim_{(\varepsilon, \delta)} \textbf{if } e_2 \textbf{ then } c_2 \textbf{ else } c_2' : \Phi \Longrightarrow \Psi}$$

$$\text{WHILE} \; \frac{\models \Phi \wedge e_v\langle 1\rangle \le 0 \rightarrow \neg e_1\langle 1\rangle \qquad \models \Phi \rightarrow e\langle 1\rangle = e\langle 2\rangle \qquad \forall K \in \mathbb{N}, \vdash c_1 \sim_{(\varepsilon, \delta)} c_2 : \Phi \wedge e_1\langle 1\rangle \wedge e_v\langle 1\rangle = K \Longrightarrow \Phi \wedge e_v\langle 1\rangle < K}{\vdash \textbf{while } e_1 \textbf{ do } c_1 \sim_{(N\varepsilon, N\delta)} \textbf{while } e_2 \textbf{ do } c_2 : \Phi \wedge e\langle 1\rangle \le N \Longrightarrow \Phi \wedge \neg e_1\langle 1\rangle}$$

Figure 4.1: Two-sided APRHL rules

Judgments and validity

Judgments in APRHL have the following form:

$$c_1 \sim_{(\varepsilon, \delta)} c_2 : \Phi \Longrightarrow \Psi$$

Just like in PRHL, Φ and Ψ are assertions on a product memory and refer to variables tagged with $\langle 1\rangle$ and $\langle 2\rangle$. The parameters ε, δ are expressions involving constants and logical variables; in particular, they do not mention program variables and do not depend on the program state.

Validity for APRHL judgments is defined in terms of approximate liftings.

Definition 4.3.1. An APRHL judgment is *valid* in logical context ρ, written

$$\rho \models c_1 \sim_{(\varepsilon, \delta)} c_2 : \Phi \Longrightarrow \Psi,$$

if for any two memories $(m_1, m_2) \in [\![\Phi]\!]_\rho$ there exists an approximate lifting relating the output distributions:

$$[\![c_1]\!]_\rho m_1 \; [\![\Psi]\!]_\rho^{\sharp([\![\varepsilon]\!]_\rho, [\![\delta]\!]_\rho)} \; [\![c_2]\!]_\rho m_2.$$

Core proof rules

Most of the rules in APRHL generalize rules from PRHL, with special handling for the (ε, δ) parameters. We present the core proof system and comment on departures from PRHL.

We begin with the two-sided rules in Fig. 4.1. The [SKIP] and [ASSN] rules are lifted from PRHL. To gain intuition for the sampling rule [LAP], we first consider a special case:

$$\text{LAP*} \;\frac{}{\vdash x \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e) \sim_{(k\varepsilon,0)} x \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e) : |e\langle 1 \rangle - e\langle 2 \rangle| \leq k \implies x\langle 1 \rangle = x\langle 2 \rangle}$$

Since the means $e\langle 1 \rangle$ and $e\langle 2 \rangle$ may not be equal, the two distributions may have different probabilities of sampling the same value and there may be no exact coupling guaranteeing $x\langle 1 \rangle = x\langle 2 \rangle$. Nevertheless, there is a $(k\varepsilon, 0)$-approximate coupling when the means differ by at most k. Since approximate lifting of equality models differential privacy, this rule captures privacy of the Laplace mechanism (Theorem 4.1.3). The full sampling rule [LAP] proves a general post-condition Ψ if it is true as a pre-condition, assuming the two sampled variables are equal.

The sequencing rule [SEQ] is similar to the sequencing rule in PRHL, summing up the approximation parameters. This rule reflects a composition principle for approximate couplings generalizing the sequential composition theorem from differential privacy (Theorem 4.1.4).

The conditional rule [COND] is similar to its counterpart from PRHL. Assuming the guards are equal initially, if there is an (ε, δ)-coupling of corresponding pairs of branches then there is an (ε, δ)-coupling of the two conditionals. Finally, the loop rule [WHILE] applies to loops that run at most a finite number of iterations N; this is enforced by the strictly decreasing integer variant e_v. Given an (ε, δ)-coupling for the loop bodies, the rule produces a $(N\varepsilon, N\delta)$-coupling of the two loops. Again, this rule corresponds to a sequential composition principle for approximate couplings.

The one-sided rules for APRHL are presented in Fig. 4.2; the structural rules, in Fig. 4.3. The one-sided sampling rules, [LAP-L] and [LAP-R], give a $(0, 0)$-lifting. The rule of consequence [CONSEQ] allows increasing the approximate parameters since larger parameters require a looser bound between the witnesses. The other rules are straightforward generalizations of their PRHL counterparts.

As expected, the logic is sound.

Theorem 4.3.2 (Soundness of APRHL). *Let ρ be a logical context. If a judgment is derivable*

$$\rho \vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \implies \Psi,$$

then it is valid:

$$\rho \models c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \implies \Psi.$$

Proof sketch. By induction on the derivation. The proof is very similar to the proof of soundness for APRHL by Olmedo (2014), with some minor adjustments to handle the special element $\star$ in our definition of approximate coupling. Appendix B gives a self-contained proof of soundness for the full logic, including the new rules we will soon introduce. □

The natural counterpart to soundness is *completeness*: valid judgments should be provable by the proof system. APRHL is incomplete in at least one respect: while valid judgments may relate commands that do not always terminate, derivable judgments can only relate lossless programs.

Lemma 4.3.3. *If $\rho \vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \implies \Psi$ is derivable, then c_1 and c_2 are both Φ-lossless.*

Proof. By induction on the derivation. Since the loop rule [WHILE] requires both loops to terminate in at most n iterations and the one-sided variants [WHILE-L]/[WHILE-R] assume losslessness, c_1 and c_2 must be lossless under the pre-condition. □

This kind of incompleteness aside, it is not known whether APRHL is complete for terminating programs (or even *relatively complete* in some natural sense); we discuss this issue further in Chapter 6.

$$\text{Assn-L} \; \frac{}{\vdash x_1 \leftarrow e_1 \sim_{(0,0)} \mathbf{skip} : \Psi \{e_1\langle 1\rangle / x_1\langle 1\rangle\} \Longrightarrow \Psi}$$

$$\text{Assn-R} \; \frac{}{\vdash \mathbf{skip} \sim_{(0,0)} x_2 \leftarrow e_2 : \Psi \{e_2\langle 2\rangle / x_2\langle 2\rangle\} \Longrightarrow \Psi}$$

$$\text{Lap-L} \; \frac{}{\vdash x_1 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_1) \sim_{(0,0)} \mathbf{skip} : \forall v \in \mathbb{Z},\ \Psi \{v/x_1\langle 1\rangle\} \Longrightarrow \Psi}$$

$$\text{Lap-R} \; \frac{}{\vdash \mathbf{skip} \sim_{(0,0)} x_2 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_2) : \forall v \in \mathbb{Z},\ \Psi \{v/x_2\langle 2\rangle\} \Longrightarrow \Psi}$$

$$\text{Cond-L} \; \frac{\vdash c_1 \sim_{(\varepsilon,\delta)} c : \Phi \wedge e_1\langle 1\rangle \Longrightarrow \Psi \qquad \vdash c_1' \sim_{(\varepsilon,\delta)} c : \Phi \wedge \neg e_1\langle 1\rangle \Longrightarrow \Psi}{\vdash \mathbf{if}\ e_1\ \mathbf{then}\ c_1\ \mathbf{else}\ c_1' \sim_{(\varepsilon,\delta)} c : \Phi \Longrightarrow \Psi}$$

$$\text{Cond-R} \; \frac{\vdash c \sim_{(\varepsilon,\delta)} c_2 : \Phi \wedge e_2\langle 2\rangle \Longrightarrow \Psi \qquad \vdash c \sim_{(\varepsilon,\delta)} c_2' : \Phi \wedge \neg e_2\langle 2\rangle \Longrightarrow \Psi}{\vdash c \sim_{(\varepsilon,\delta)} \mathbf{if}\ e_2\ \mathbf{then}\ c_2\ \mathbf{else}\ c_2' : \Phi \Longrightarrow \Psi}$$

$$\text{While-L} \; \frac{\vdash c_1 \sim_{(0,0)} \mathbf{skip} : \Phi \wedge e_1\langle 1\rangle \Longrightarrow \Phi \qquad \models \Phi \rightarrow \Phi_1\langle 1\rangle \qquad \Phi_1 \models \mathbf{while}\ e_1\ \mathbf{do}\ c_1\ \text{lossless}}{\vdash \mathbf{while}\ e_1\ \mathbf{do}\ c_1 \sim_{(0,0)} \mathbf{skip} : \Phi \Longrightarrow \Phi \wedge \neg e_1\langle 1\rangle}$$

$$\text{While-R} \; \frac{\vdash \mathbf{skip} \sim_{(0,0)} c_2 : \Phi \wedge e_2\langle 2\rangle \Longrightarrow \Phi \qquad \models \Phi \rightarrow \Phi_2\langle 2\rangle \qquad \Phi_2 \models \mathbf{while}\ e_2\ \mathbf{do}\ c_2\ \text{lossless}}{\vdash \mathbf{skip} \sim_{(0,0)} \mathbf{while}\ e_2\ \mathbf{do}\ c_2 : \Phi \Longrightarrow \Phi \wedge \neg e_2\langle 2\rangle}$$

Figure 4.2: One-sided APRHL rules

4.4 Proof by approximate coupling

Much like PRHL is a logic for formal proofs by coupling, APRHL can be viewed as a logic for formal *proofs by approximate coupling*. With the logical rules in hand, we can work out an intuitive understanding of these proofs.

First of all, the close resemblance between PRHL and APRHL indicates that proofs by approximate couplings are broadly similar to proofs by coupling; the sampling rule [LAP] shows we can choose an approximate coupling for sampling statements (although for the moment we have just one choice), the sequencing rule [SEQ] indicates that we can sequence approximate couplings together, and the case rule [CASE] lets us select an approximate coupling based on the current state of the coupled executions.

The main difference is we must track the approximation parameters ε and δ as we build the coupling. When we apply the sampling rule [LAP], for instance, we accrue parameters $(k \cdot \varepsilon, 0)$ where k is an upper bound on the distance between the means. In the sequencing rule [SEQ] (and similarly in the loop rule [WHILE]), we add up the approximate couplings parameters for the sequenced commands. The resulting style of analysis blends proof by coupling with the cost interpretation of differential privacy (Remark 4.1.6). For instance, we can think of the rule [LAP] as paying for the privacy cost to couple the samples to be equal. Accordingly, proofs by approximate coupling recover proofs by the standard composition theorem (Theorem 4.1.4). By introducing other approximate

$$\text{CONSEQ} \; \frac{\vdash c_1 \sim_{(\varepsilon',\delta')} c_2 : \Phi' \Longrightarrow \Psi' \quad \models \Phi \to \Phi' \quad \models \Psi' \to \Psi \quad \models \varepsilon' \leq \varepsilon \quad \models \delta' \leq \delta}{\vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi}$$

$$\text{EQUIV} \; \frac{\vdash c_1' \sim_{(\varepsilon,\delta)} c_2' : \Phi \Longrightarrow \Psi \quad c_1 \equiv c_1' \quad c_2 \equiv c_2'}{\vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi}$$

$$\text{CASE} \; \frac{\vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \wedge \Theta \Longrightarrow \Psi \quad \vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \wedge \neg\Theta \Longrightarrow \Psi}{\vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi}$$

$$\text{TRANS} \; \frac{\vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi \quad \vdash c_2 \sim_{(\varepsilon',\delta')} c_3 : \Phi' \Longrightarrow \Psi'}{\vdash c_1 \sim_{(\varepsilon+\varepsilon',\exp(\varepsilon')\delta+\delta')} c_3 : \Phi' \circ \Phi \Longrightarrow \Psi' \circ \Psi}$$

$$\text{FRAME} \; \frac{\vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi \quad \text{FV}(\Theta) \cap \text{MV}(c_1, c_2) = \varnothing}{\vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \wedge \Theta \Longrightarrow \Psi \wedge \Theta}$$

Figure 4.3: Structural APRHL rules

$$\text{LAPNULL} \; \frac{\begin{array}{c} x_1, x_2 \notin \text{FV}(e_1, e_2) \\ \Phi \triangleq \forall w_1, w_2 \in \mathbb{Z}, \; w_1 - w_2 = e_1\langle 1\rangle - e_2\langle 2\rangle \to \Psi\{w_1, w_2/x_1\langle 1\rangle, x_2\langle 2\rangle\} \end{array}}{\vdash x_1 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_1) \sim_{(0,0)} x_2 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_2) : \Phi \Longrightarrow \Psi}$$

$$\text{LAPGEN} \; \frac{\begin{array}{c} x_1, x_2 \notin \text{FV}(e_1, e_2) \\ \Phi \triangleq |k + e_1\langle 1\rangle - e_2\langle 2\rangle| \leq k' \wedge \forall w_1, w_2 \in \mathbb{Z}, \; w_1 + k = w_2 \to \Psi\{w_1, w_2/x_1\langle 1\rangle, x_2\langle 2\rangle\} \end{array}}{\vdash x_1 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_1) \sim_{(k'\cdot\varepsilon,0)} x_2 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_2) : \Phi \Longrightarrow \Psi}$$

Figure 4.4: New Laplace rules for APRHL

couplings for the Laplace distribution, we can achieve clean and compositional approximate coupling proofs of privacy even when the standard composition theorem from differential privacy does not suffice.

4.5   New couplings for the Laplace distribution

Unlike the rule [SAMPLE] in PRHL, which can couple two uniform distributions in different ways by varying the bijection, the Laplace rule [LAP] can only couple samples to be equal. To support richer proofs, we introduce two new approximate couplings for the Laplace distribution and build them into APRHL rules.

Null coupling

Suppose we want to couple the Laplace distributions $\mathbf{Lap}_\varepsilon(v_1)$ and $\mathbf{Lap}_\varepsilon(v_2)$. Sampling from these distributions is equivalent to sampling from $\mathbf{Lap}_\varepsilon(0)$ and then adding v_1 and v_2 respectively, so we can couple by using equal draws from $\mathbf{Lap}_\varepsilon(0)$. Since equal draws from the same distribution have the same probability, this approximate

coupling is in fact an exact, $(0,0)$-coupling, an analogue of the identity coupling (Proposition 2.1.10). More formally, we have the following result.

Proposition 4.5.1. *Let $v_1, v_2 \in \mathbb{Z}$. Then:*

$$\mathbf{Lap}_\varepsilon(v_1) \, \{(x_1, x_2) \mid x_1 - v_1 = x_2 - v_2\}^{\sharp(0,0)} \, \mathbf{Lap}_\varepsilon(v_2).$$

Proof. We construct witnesses $\mu_L, \mu_R \in \mathbf{Distr}(\mathbb{Z}^\star \times \mathbb{Z}^\star)$. Define the relation

$$\mathcal{R} \triangleq \{(x_1, x_2) \in \mathbb{Z} \times \mathbb{Z} \mid x_1 - v_1 = x_2 - v_2\}$$

and let $L(r)$ be probability $\mathbf{Lap}_\varepsilon(0)$ produces r. Define witnesses

$$\mu_L(x_1, x_2) = \mu_R(x_1, x_2) \triangleq \begin{cases} L(x_1 - v_1) & : (x_1, x_2) \in \mathcal{R} \\ 0 & : \text{otherwise.} \end{cases}$$

Since $\mu_L = \mu_R$, we know $d_0(\mu_L, \mu_R) = 0$. Also, $\mathrm{supp}(\mu_L)$ and $\mathrm{supp}(\mu_R)$ lie in $\mathcal{R} \subseteq \mathcal{R}^\star$. So, it remains to check the marginal conditions. Using the support condition, we have

$$\pi_1(\mu_L)(r) = \mu_L(r, r - v_1 + v_2) = L(r - v_1) = \mathbf{Lap}_\varepsilon(v_1)(r).$$

A similar calculation shows $\pi_2(\mu_R) = \mathbf{Lap}_\varepsilon(v_2)$, so μ_L and μ_R witness the approximate coupling. $\qquad \square$

We can capture this approximate coupling with the rule [LAPNULL] in Fig. 4.4. To gain intuition, the following rule is a simplified special case:

$$\text{LAPNULL*} \; \frac{x \notin \mathrm{FV}(e)}{\vdash x \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e) \sim_{(0,0)} x \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e) : \top \implies x\langle 1 \rangle - x\langle 2 \rangle = e\langle 1 \rangle - e\langle 2 \rangle}$$

The coupling keeps the distance between the samples the same as the distance between the means. The general rule [LAPNULL] can prove post-conditions of any shape.

Theorem 4.5.2. *The rule [LAPNULL] is sound.*

Proof. We leave the logical context ρ implicit. Let $V \triangleq \mathcal{X} \setminus \{x_1, x_2\}$ be the non-sampled variables; we write $m[V]$ for the restriction of a memory m to variables in V. Consider any two memories m_1, m_2, let the means be $v_1 \triangleq \llbracket e_1 \rrbracket m_1$ and $v_2 \triangleq \llbracket e_2 \rrbracket m_2$, and let the output distributions be

$$\mu_1 \triangleq \llbracket x_1 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_1) \rrbracket m_1 \quad \text{and} \quad \mu_2 \triangleq \llbracket x_2 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_2) \rrbracket m_2.$$

We construct an approximate coupling between μ_1 and μ_2. By Proposition 4.5.1 we have

$$\mathbf{Lap}_\varepsilon(v_1) \, \{(x_1, x_2) \mid x_1 - v_1 = x_2 - v_2\}^{\sharp(0,0)} \, \mathbf{Lap}_\varepsilon(v_2).$$

By Theorem 4.2.7 with maps $\llbracket x_1 \rrbracket$ and $\llbracket x_2 \rrbracket$, we obtain

$$\mu_1 \, \llbracket x_1\langle 1 \rangle - v_1 = x_2\langle 2 \rangle - v_2 \rrbracket^{\sharp(0,0)} \, \mu_2.$$

By the free variable condition, $v_1 = \llbracket e_1 \rrbracket m_1'$ and $v_2 = \llbracket e_2 \rrbracket m_2'$ for every memory $m_1' \in \mathrm{supp}(\mu_1)$ and $m_2' \in \mathrm{supp}(\mu_2)$, so we may assume by Proposition 4.2.6 that the witnesses are supported on such memories, giving witnesses to

$$\mu_1 \, \llbracket x_1\langle 1 \rangle - e_1\langle 1 \rangle = x_2\langle 2 \rangle - e_2\langle 2 \rangle \rrbracket^{\sharp(0,0)} \, \mu_2.$$

Also by the free variable condition, $m_1'[V] = m_1[V]$ and $m_2'[V] = m_2[V]$ so

$$\mu_1 \ \{(m_1', m_2') \mid m_1'[V] = m_1[V],\ m_2'[V] = m_2[V],\ m_1'(x_1) - [\![e_1]\!]m_1 = m_2'(x_2) - [\![e_2]\!]m_2\}^{\sharp(0,0)} \ \mu_2.$$

By the pre-condition, (m_1, m_2) satisfy

$$\forall w_1, w_2 \in \mathbb{Z},\ w_1 - w_2 = e_1\langle 1 \rangle - e_2\langle 2 \rangle \to \Psi\ \{w_1, w_2/x_1\langle 1 \rangle, x_2\langle 2 \rangle\}$$

and so

$$\mu_1 \ \Psi^{\sharp(0,0)} \ \mu_2,$$

showing [LapNull] is sound. □

General coupling

Our next approximate coupling shifts the samples apart by a constant amount. Suppose we want to approximately couple the Laplace distributions $\mathbf{Lap}_\varepsilon(v_1)$ and $\mathbf{Lap}_\varepsilon(v_2)$ so that the samples x_1, x_2 satisfy $x_1 + k = x_2$. Intuitively, the approximation parameters should depend on the shift k and the distance $|v_1 - v_2|$ between the means—larger shifts and larger distances imply that we match samples with greater difference in probabilities. More formally, we have the following result.

Proposition 4.5.3. *Let $k, k', v_1, v_2 \in \mathbb{Z}$, and suppose $|k + v_1 - v_2| \le k'$. Then:*

$$\mathbf{Lap}_\varepsilon(v_1) \ \{(x_1, x_2) \mid x_1 + k = x_2\}^{\sharp(k'\varepsilon, 0)} \ \mathbf{Lap}_\varepsilon(v_2).$$

Proof. We need two witnesses $\mu_L, \mu_R \in \mathbf{Distr}(\mathbb{Z}^\star \times \mathbb{Z}^\star)$. Define the relation

$$\mathcal{R} \triangleq \{(x_1, x_2) \in \mathbb{Z} \times \mathbb{Z} \mid x_1 + k = x_2\}$$

and let $L(r)$ be the probability $\mathbf{Lap}_\varepsilon(0)$ produces r. Define witnesses

$$\mu_L(x_1, x_2) \triangleq \begin{cases} L(x_1 - v_1) : (x_1, x_2) \in \mathcal{R} \\ 0 : \text{otherwise} \end{cases} \quad \text{and} \quad \mu_R(x_1, x_2) \triangleq \begin{cases} L(x_2 - v_2) : (x_1, x_2) \in \mathcal{R} \\ 0 : \text{otherwise}. \end{cases}$$

By definition, $\text{supp}(\mu_L)$ and $\text{supp}(\mu_R)$ lie in $\mathcal{R} \subseteq \mathcal{R}^\star$. The marginal conditions are easy to check. So, it remains to check the distance condition. It suffices to show

$$\mu_L(x_1, x_2) \le \exp(k'\varepsilon) \cdot \mu_R(x_1, x_2)$$

for every $(x_1, x_2) \in \mathbb{Z}^\star \times \mathbb{Z}^\star$, since summing over any set $\mathcal{S} \subseteq \mathbb{Z}^\star \times \mathbb{Z}^\star$ gives $\mu_L(\mathcal{S}) \le \exp(k'\varepsilon) \cdot \mu_R(\mathcal{S})$.

Clearly the claim is true for $(x_1, x_2) \notin \mathcal{R}$; note that μ_L and μ_R are both zero when x_1 or x_2 is $\star$. Otherwise we just need to bound

$$L(x_1 - v_1) \le \exp(k'\varepsilon) \cdot L(x_2 - v_2)$$

where $x_1 + k = x_2$. Unfolding definitions, it suffices to bound

$$\exp(-|x_1 - v_1|\varepsilon) \le \exp(k'\varepsilon) \cdot \exp(-|x_1 + k - v_2|\varepsilon),$$

which follows by assumption:

$$|x_1 + k - v_2| - |x_1 - v_1| \le |k - v_2 + v_1| \le k'.$$

So, $d_{k'\varepsilon}(\mu_L, \mu_R) \le 0$ and μ_L, μ_R witness the approximate coupling. □

This approximate coupling is modeled by the rule [LAPGEN], in Fig. 4.4. Note that k and k' must be logical expressions, independent of the program state. To gain intuition, the following rule is a simplified special case:

$$\text{LAPGEN*} \quad \frac{}{\vdash x \overset{\$}{\leftarrow} \mathbf{Lap}_\varepsilon(e) \sim_{(k'\cdot\varepsilon,0)} x \overset{\$}{\leftarrow} \mathbf{Lap}_\varepsilon(e) : |k + e\langle 1\rangle - e\langle 2\rangle| \leq k' \implies x\langle 1\rangle + k = x\langle 2\rangle}$$

As expected, the post-condition ensures that the coupled samples are shifted apart by k. The approximation parameters scale as k'; this measures the difference between the k-shifted means. As a sanity check, when $k' = 0$ the distribution means are shifted by k and we have an exact, $(0,0)$-coupling. The general rule [LAPGEN] can prove post-conditions of any shape.

Theorem 4.5.4. *The rule* [LAPGEN] *is sound.*

Proof. We leave the logical context ρ implicit. Let $V \triangleq \mathcal{X} \setminus \{x_1, x_2\}$ be the non-sampled variables; we write $m[V]$ for the restriction of a memory m to variables in V. Consider any two memories m_1, m_2, let the means be $v_1 \triangleq [\![e_1]\!]m_1$ and $v_2 \triangleq [\![e_2]\!]m_2$ such that $|k + v_1 - v_2| \leq k'$, and let the output distributions be

$$\mu_1 \triangleq [\![x_1 \overset{\$}{\leftarrow} \mathbf{Lap}_\varepsilon(e_1)]\!]m_1 \quad \text{and} \quad \mu_2 \triangleq [\![x_2 \overset{\$}{\leftarrow} \mathbf{Lap}_\varepsilon(e_2)]\!]m_2.$$

We construct an approximate coupling between μ_1 and μ_2. By Proposition 4.5.3, we have

$$\mathbf{Lap}_\varepsilon(v_1) \{(x_1, x_2) \mid x_1 + k = x_2\}^{\sharp(k'\varepsilon, 0)} \mathbf{Lap}_\varepsilon(v_2).$$

By Theorem 4.2.7 with maps $[\![x_1]\!]$ and $[\![x_2]\!]$, we obtain

$$\mu_1 [\![x_1\langle 1\rangle + k = x_2\langle 2\rangle]\!]_\rho^{\sharp(k'\varepsilon, 0)} \mu_2.$$

By the free variable condition, $m_1'[V] = m_1[V]$ and $m_2'[V] = m_2[V]$ for all memories $m_1' \in \text{supp}(\mu_1)$ and $m_2' \in \text{supp}(\mu_2)$, so we may assume by Proposition 4.2.6 that the witnesses are supported on such memories. Hence, we have the following lifting:

$$\mu_1 \{(m_1', m_2') \mid m_1'[V] = m_1[V], \ m_2'[V] = m_2[V], \ m_1'(x_1) + k = m_2'(x_2) \in [\![x_1\langle 1\rangle + k = x_2\langle 2\rangle]\!]\}^{\sharp(k'\varepsilon, 0)} \mu_2$$

By the pre-condition, (m_1, m_2) satisfy

$$\forall w_1, w_2 \in \mathbb{Z}, \ w_1 + k = w_2 \rightarrow \Psi \{w_1, w_2 / x_1\langle 1\rangle, x_2\langle 2\rangle\}$$

and so

$$\mu_1 \Psi^{\sharp(k'\varepsilon, 0)} \mu_2,$$

showing [LAPGEN] is sound. $\qquad\square$

Remark 4.5.5. If we could take $k' \triangleq 0$ and $k \triangleq e_2\langle 2\rangle - e_1\langle 1\rangle$ in [LAPGEN], we would recover [LAPNULL]. However, k must be a constant or logical variable. (We will discuss possible ways to lift this requirement in Section 6.1.)

4.6 Pointwise privacy

In sophisticated privacy proofs, it is often convenient to focus on a single output at a time. We call this pattern *pointwise equality* and formalize it as the following property of approximate couplings.

$$\text{PW-EQ} \ \dfrac{\begin{array}{c} \models \sum_{i \in \mathcal{R}} \delta'\,\{i/\gamma\} \leq \delta \qquad \gamma \notin \text{FV}(\Phi, c_1, c_2, e_1, e_2, \varepsilon, \delta) \\ \forall \gamma \in \mathcal{R}, \ \vdash c_1 \sim_{(\varepsilon, \delta')} c_2 : \Phi \Longrightarrow e_1\langle 1 \rangle = \gamma \to e_2\langle 2 \rangle = \gamma \end{array}}{\vdash c_1 \sim_{(\varepsilon, \delta)} c_2 : \Phi \Longrightarrow e_1\langle 1 \rangle = e_2\langle 2 \rangle}$$

Figure 4.5: Pointwise equality rule [PW-EQ] for APRHL

Proposition 4.6.1. *Let μ_1, μ_2 be sub-distributions over $\mathcal{R}$ and suppose for every $i \in \mathcal{R}$, we have*

$$\mu_1 \ \{(r_1, r_2) \mid r_1 = i \to r_2 = i\}^{\sharp(\varepsilon, \delta_i)} \ \mu_2$$

for non-negative ε and $\{\delta_i\}_{i \in \mathcal{R}}$. Then we have

$$\mu_1 \ (=)^{\sharp(\varepsilon, \delta)} \ \mu_2$$

where $\delta = \sum_{i \in \mathcal{R}} \delta_i$.

Proof. By Proposition 4.2.4 we know for every $i \in \mathcal{R}$,

$$\mu_1(i) \leq \exp(\varepsilon) \cdot \mu_2(i) + \delta_i.$$

So for any subset $\mathcal{S} \subseteq \mathcal{R}$, summing over $i \in \mathcal{S}$ yields

$$\mu_1(\mathcal{S}) \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{S}) + \sum_{i \in \mathcal{S}} \delta_i \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{S}) + \delta$$

since $\delta_i \geq 0$. We define witnesses $\mu_L(r, r) \triangleq \mu_1(r)$ and $\mu_R(r, r) \triangleq \mu_2(r)$ for $r \neq \star$, and zero otherwise. The support and marginal conditions are easy to check. For the distance condition, consider any set $\mathcal{T} \subseteq \mathcal{R}^\star \times \mathcal{R}^\star$ and let

$$\mathcal{T}' \triangleq \mathcal{T} \cap \{(r_1, r_2) \in \mathcal{R} \times \mathcal{R} \mid r_1 = r_2\}.$$

We know $\mu_L(\mathcal{T}) = \mu_L(\mathcal{T}')$ and $\mu_R(\mathcal{T}) = \mu_R(\mathcal{T}')$. Letting $\mathcal{S}' = \{r \in \mathcal{R} \mid (r, r) \in \mathcal{T}'\}$, we have

$$\mu_L(\mathcal{T}') = \mu_1(\mathcal{S}') \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{S}') + \delta \leq \exp(\varepsilon) \cdot \mu_R(\mathcal{T}') + \delta$$

so $d_\varepsilon(\mu_L, \mu_R) \leq \delta$ and we have witnesses as desired.   □

Pointwise equality simplifies coupling proofs of differential privacy: rather than proving differential privacy in one shot, we can give a separate proof for each possible output and then combine the results. We can internalize pointwise equality as the APRHL rule [PW-EQ] in Fig. 4.5. In the premise, the pointwise judgment is indexed by a logical variable γ. The conclusion gives an approximate lifting of equality in the post-condition.

Theorem 4.6.2. *The rule [PW-EQ] is sound.*

Proof. Let ρ be the logical context. The proof follows essentially by Proposition 4.6.1, handling the logical variables carefully. Consider two memories $(m_1, m_2) \in [\![\Phi]\!]_\rho$ and output distributions

$$\mu_1 \triangleq [\![c_1]\!]_\rho m_1 \quad \text{and} \quad \mu_2 \triangleq [\![c_2]\!]_\rho m_2.$$

We construct an approximate lifting relating μ_1 and μ_2. By the free variable condition, $(m_1, m_2) \in \llbracket \Phi \rrbracket_{\rho \cup \gamma \mapsto i}$ for any i and so by validity of the premises, we can use the forward direction of Theorem 4.2.7 to project the liftings in the premises to the expressions e_1 and e_2:

$$(\llbracket e_1 \rrbracket_{\rho \cup \gamma \mapsto i})^\sharp (\mu_1) \, \{(a_1, a_2) \in \mathcal{R} \times \mathcal{R} \mid a_1 = i \to a_2 = i\}^{\sharp(\varepsilon, \delta')} \, (\llbracket e_2 \rrbracket_{\rho \cup \gamma \mapsto i})^\sharp (\mu_2)$$

for each $i \in \mathcal{R}$. (Technically ε and δ' are also interpreted in the logical context $\rho \cup \gamma \mapsto i$; we elide this.) By the free variable condition, the two projected distributions are in fact the same for all i, and everything besides δ' can be interpreted in the original context ρ. Proposition 4.6.1 with $\delta_i \triangleq \llbracket \delta' \rrbracket_{\rho \cup \gamma \mapsto i}$ gives

$$\llbracket e_1 \rrbracket_\rho^\sharp (\mu_1) \, \{(a_1, a_2) \mid a_1 = a_2\}^{\sharp(\varepsilon, \delta)} \, \llbracket e_2 \rrbracket_\rho^\sharp (\mu_2),$$

and the reverse direction of Theorem 4.2.7 with maps $\llbracket e_1 \rrbracket_\rho$ and $\llbracket e_2 \rrbracket_\rho$ gives

$$\mu_1 \, (\llbracket e_1 \langle 1 \rangle = e_2 \langle 2 \rangle \rrbracket_\rho)^{\sharp(\varepsilon, \delta)} \, \mu_2.$$

Thus, [PW-EQ] is sound. □

Remark 4.6.3. From a logical perspective, pointwise equality resembles the *Leibniz equality* principle:

$$\models (\forall i, \, x = i \to y = i) \to x = y.$$

Indeed, if APRHL had a structural rule to convert an external universal quantifier into an internal universal quantifier, e.g., something like

$$\text{FORALL} \quad \frac{\forall i, \, \vdash c_1 \sim_{(\varepsilon, \delta)} c_2 : \Phi \implies \Psi_i}{\vdash c_1 \sim_{(\varepsilon, \delta)} c_2 : \Phi \implies \forall i, \, \Psi_i}$$

[PW-EQ] could be derived using the rule of consequence with Leibniz equality. Unfortunately this rule is not sound, not even in PRHL. In fact, if we have just two judgments with post-conditions Ψ_1 and Ψ_2, it is not sound in general to combine them into a single judgment with post-condition $\Psi_1 \wedge \Psi_2$: the underlying witnesses may be different. The rule [PW-EQ] is a special case where we may safely combine post-conditions across different judgments.

Remark 4.6.4. On a more practical note, the post-condition in [PW-EQ] is highly specific—the assertion must be equality. In Chapter 5 we will see some ways to partially mitigate this limitation, for instance by incorporating one-sided conjuncts (Section 5.2).

4.7 Coupling proofs of privacy

Approximate coupling proofs are a convenient and compositional tool for proving differential privacy. Starting from two adjacent inputs, we select an approximate coupling for each pair of corresponding sampling instructions such that (i) the total cost does not exceed the target privacy parameters (ε, δ), and (ii) the outputs on the two executions are equal under the approximate coupling. By pointwise equality, we can sometimes establish point (ii) by building an approximate coupling separately for each possible output value i, ensuring that if the first output is equal to i then the second output is also equal to i. We will apply the asymmetric approximate couplings from Section 4.5 to induce this kind of asymmetric relation on outputs.

Compared to existing proofs of privacy, approximate coupling proofs are simpler and more concise, abstracting away reasoning about conditional probabilities. To demonstrate, we prove differential privacy for two algorithms from the privacy literature. We present each proof twice: first as an approximate coupling proof, then as a formal proof in APRHL.

The Report-noisy-max mechanism

Our first example is called *Report-noisy-max* (Dwork and Roth, 2014). Given a list of numeric queries $q_1, \ldots, q_N :$ $\mathcal{D} \to \mathbb{Z}$ and a database $d \in \mathcal{D}$, this mechanism computes $q_i(d)$ for each i and adds fresh Laplace noise to each answer, releasing the index i with the highest noisy answer. Alternatively, we can think of each query as indexed by an element r in some finite range $\mathcal{R}$, where q_r computes the *score* for r given private data d. Then Report-noisy-max is a close cousin to the well-known *Exponential mechanism* of McSherry and Talwar (2007), which finds an element with a high score while preserving privacy.

Suppose **evalQ**(i, d) returns $q_i(d)$. We implement Report-noisy-max as the following program *rnm*:

$$maxA \leftarrow 0;$$
$$maxI \leftarrow 0;$$
$$i \leftarrow 1;$$
$$\textbf{while } i \leq N \textbf{ do}$$
$$\quad a \xleftarrow{\$} \textbf{Lap}_{\varepsilon/2}(\textbf{evalQ}(i, d));$$
$$\quad \textbf{if } maxI = 0 \vee a > maxA \textbf{ then}$$
$$\quad\quad maxA \leftarrow a; maxI \leftarrow i;$$
$$\quad i \leftarrow i + 1$$

The variable *maxI* stores the output of the mechanism; we assume it ranges over $\mathbb{N}$.

Theorem 4.7.1. *Suppose each query q_i is 1-sensitive: $|q_i(d) - q_i(d')| \leq 1$ for adjacent databases d, d'. Then executing rnm and returning maxI is ε-differentially private.*

While we could prove privacy with the sequential composition theorem (Theorem 4.1.4), we would get an overly conservative bound of $(N\varepsilon, 0)$-privacy since we must pay for each Laplace sampling. Report-noisy-max is an example of a mechanism where the precise analysis showing $(\varepsilon, 0)$-privacy previously required an ad hoc proof. However, since approximate couplings satisfy a more general composition principle, we can prove privacy for this mechanism compositionally.

Proof by approximate coupling. Consider a possible output $j \in \mathbb{N}$. We construct an $(\varepsilon, 0)$-approximate coupling such that if $maxI\langle 1 \rangle = j$, then $maxI\langle 2 \rangle = j$. If $j = 0$ this is easy since the only way $maxI = 0$ is if $N = 0$ and the loops terminate immediately. If $j > N$ this is also easy, as $maxI$ cannot exceed N.

So suppose $j \in [1, N]$. In iterations $i \neq j$, we couple the samplings so both runs use the same amount of noise:

$$a\langle 1 \rangle - \textbf{evalQ}(i\langle 1 \rangle, d\langle 1 \rangle) = a\langle 2 \rangle - \textbf{evalQ}(i\langle 2 \rangle, d\langle 2 \rangle).$$

In particular, $a\langle 2 \rangle \leq a\langle 1 \rangle + 1$. This is a $(0, 0)$-approximate coupling for each iteration. For iteration $i = j$, we couple so the noisy answer in the second run is one larger than the noisy answer in the first run:

$$a\langle 1 \rangle + 1 = a\langle 2 \rangle.$$

The true answers **evalQ**$(i\langle 1 \rangle, d\langle 1 \rangle)$ and **evalQ**$(i\langle 2 \rangle, d\langle 2 \rangle)$ are at most 1 apart and the shift is 1. Since we use Laplace noise with parameter $\varepsilon/2$, this is a $(2 \cdot \varepsilon/2, 0) = (\varepsilon, 0)$-coupling.

Now if the noisy answer on iteration j is the highest noisy answer in the first run, then it must also be the highest noisy answer in the second run: by the coupling, $a\langle 1 \rangle + 1 = a\langle 2 \rangle$ for iteration j and $a\langle 2 \rangle \leq a\langle 1 \rangle + 1$ for all other iterations. The total cost is $(\varepsilon, 0)$, establishing $(\varepsilon, 0)$-differential privacy. $\qquad\square$

Remark 4.7.2. Earlier versions of Report-noisy-max also returned the noisy answer *maxA* in addition to the index *maxI*. However, subtle errors in the privacy proof were later discovered; a correct proof of privacy is currently not known. Attempting a proof by approximate coupling immediately runs into a problem: we have coupled $a\langle 1 \rangle + 1 = a\langle 2 \rangle$ for the critical iteration, but we need $a\langle 1 \rangle = a\langle 2 \rangle$ if we are to safely return the noisy answer too.

In order to perform this proof in APRHL, the main complication is to only pay for coupling the critical iteration j. Directly applying the loop rule would give an overly conservative guarantee of $(N\varepsilon, 0)$-privacy since [WHILE] assumes each iteration has the same cost. To get around this problem, we first use the program equivalence rule to split the loop into three separate pieces: iterations before j, iteration j, and iterations after j. Then we arrange a $(0, 0)$-coupling for each iteration in the first and last loops, and an $(\varepsilon, 0)$-coupling for the middle loop consisting of just the critical iteration.

Theorem 4.7.3. *Suppose each query q_i is 1-sensitive: $|q_i(d) - q_i(d')| \leq 1$ for adjacent databases d, d'. Then the following judgment is derivable in APRHL:*

$$\vdash rnm \sim_{(\varepsilon,0)} rnm : Adj(d\langle 1\rangle, d\langle 2\rangle) \implies maxI\langle 1\rangle = maxI\langle 2\rangle$$

Proof. We verify an equivalent program, dividing the loop in three:

$$
\begin{aligned}
&maxA \leftarrow 0; \\
&maxI \leftarrow 0; \\
&i \leftarrow 1; \\
&\textbf{while } i \leq N \wedge i < j \textbf{ do} \\
&\quad a \xleftarrow{\$} \textbf{Lap}_{\varepsilon/2}(\textbf{evalQ}(i, d)); \\
&\quad \textbf{if } maxI = 0 \vee a > maxA \textbf{ then} \\
&\quad\quad maxA \leftarrow a; maxI \leftarrow i; \\
&\quad i \leftarrow i + 1; \\
&\textbf{while } i \leq N \wedge i = j \textbf{ do} \\
&\quad a \xleftarrow{\$} \textbf{Lap}_{\varepsilon/2}(\textbf{evalQ}(i, d)); \\
&\quad \textbf{if } maxI = 0 \vee a > maxA \textbf{ then} \\
&\quad\quad maxA \leftarrow a; maxI \leftarrow i; \\
&\quad i \leftarrow i + 1; \\
&\textbf{while } i \leq N \textbf{ do} \\
&\quad a \xleftarrow{\$} \textbf{Lap}_{\varepsilon/2}(\textbf{evalQ}(i, d)); \\
&\quad \textbf{if } maxI = 0 \vee a > maxA \textbf{ then} \\
&\quad\quad maxA \leftarrow a; maxI \leftarrow i; \\
&\quad i \leftarrow i + 1
\end{aligned}
$$

We call this program rnm'. Our goal is to prove the pointwise judgment

$$\vdash rnm' \sim_{(\varepsilon,0)} rnm' : Adj(d\langle 1\rangle, d\langle 2\rangle) \implies maxI\langle 1\rangle = j \rightarrow maxI\langle 2\rangle = j$$

for every $j \in \mathbb{N}$. When $j = 0$ or $j > N$, the proof is straightforward—in the first case $N = 0$, and in the second case $maxI\langle 1\rangle = j$ must be false. So we focus on the more interesting cases, $j \in [1, N]$. The initial assignment statements can be handled with [ASSN]. Let the three loops be $w_<$, $w_=$, and $w_>$, and let *body* be the common loop body. Define the following invariants:

$$
\Theta_< \triangleq \begin{cases}
|maxA\langle 1\rangle - maxA\langle 2\rangle| \leq 1 \\
maxI\langle 1\rangle < i\langle 1\rangle \wedge maxI\langle 2\rangle < i\langle 1\rangle \\
\neg(i\langle 1\rangle \leq N \wedge i\langle 1\rangle < j) \rightarrow i\langle 1\rangle = j
\end{cases}
$$

$$
\Theta_= \triangleq \begin{cases}
|maxA\langle 1\rangle - maxA\langle 2\rangle| \leq 1 \\
maxI\langle 1\rangle = j \rightarrow (maxI\langle 2\rangle = j \wedge maxA\langle 1\rangle + 1 = maxA\langle 2\rangle) \\
\neg(i\langle 1\rangle \leq N \wedge i\langle 1\rangle = j) \rightarrow i\langle 1\rangle = j + 1
\end{cases}
$$

$$
\Theta_> \triangleq \begin{cases}
i\langle 1\rangle > j \\
maxI\langle 1\rangle = j \rightarrow (maxI\langle 2\rangle = j \wedge maxA\langle 1\rangle + 1 = maxA\langle 2\rangle)
\end{cases}
$$

We leave the invariant $Adj(d\langle 1\rangle, d\langle 2\rangle) \wedge i\langle 1\rangle = i\langle 2\rangle$ implicit and we prove three judgments corresponding to the three cases. First, we have

$$\vdash body \sim_{(0,0)} body : \Theta_< \Longrightarrow \Theta_<$$

by coupling the Laplace samplings using [LAPNULL], ensuring $|maxA\langle 1\rangle - maxA\langle 2\rangle| \leq 1$. Thus, we have the following judgment for the first loop by [WHILE]:

$$\vdash w_< \sim_{(0,0)} w_< : \Theta_< \Longrightarrow \Theta_< \wedge \neg(i\langle 1\rangle \leq N \wedge i\langle 1\rangle < j).$$

For the next loop body, we have

$$\vdash body \sim_{(\varepsilon,0)} body : \Theta_= \Longrightarrow \Theta_=$$

by coupling the Laplace samplings using [LAPGEN] with $k = 1, k' = 2$, ensuring $a\langle 1\rangle + 1 = a\langle 2\rangle$. Combined with the pre-condition $\Theta_=$, if the first run exceeds $maxA\langle 1\rangle$ then the second run also exceeds $maxA\langle 2\rangle$. By the rule [LAPGEN], this coupling costs $(\varepsilon, 0)$. Since this loop runs for just one iteration, we have a judgment for the second loop by [WHILE]:

$$\vdash w_= \sim_{(\varepsilon,0)} w_= : \Theta_= \Longrightarrow \Theta_= \wedge \neg(i\langle 1\rangle \leq N \wedge i\langle 1\rangle = j).$$

Finally for the last loop, we have

$$\vdash body \sim_{(0,0)} body : \Theta_> \Longrightarrow \Theta_>$$

by coupling the samplings using [LAPNULL]. Applying [WHILE] gives a similar judgment for the last loop:

$$\vdash w_> \sim_{(0,0)} w_> : \Theta_> \Longrightarrow \Theta_>$$

We can combine the three loop judgments while summing the approximation parameters with [SEQ], applying the rule of consequence with implications

$$\models \Theta_< \wedge \neg(i\langle 1\rangle \leq N \wedge i\langle 1\rangle < j) \rightarrow \Theta_=$$
$$\models \Theta_= \wedge \neg(i\langle 1\rangle \leq N \wedge i\langle 1\rangle = j) \rightarrow \Theta_>$$

to establish

$$\vdash rnm' \sim_{(\varepsilon,0)} rnm' : Adj(d\langle 1\rangle, d\langle 2\rangle) \Longrightarrow maxI\langle 1\rangle = j \rightarrow maxI\langle 2\rangle = j.$$

We conclude differential privacy by applying [PW-EQ] and [EQUIV]:

$$\vdash rnm \sim_{(\varepsilon,0)} rnm : Adj(d\langle 1\rangle, d\langle 2\rangle) \Longrightarrow maxI\langle 1\rangle = maxI\langle 2\rangle. \qquad \square$$

Remark 4.7.4. Report-noisy-max draws noise from the Laplace distribution. A slight variant of this algorithm uses the *one-sided Laplace* distribution, also called the *exponential* distribution, to achieve similar results. This variant is closely related to the Exponential mechanism of McSherry and Talwar (2007); for instance, if we add noise from the continuous exponential distribution, Report-noisy-max is equivalent to the Exponential mechanism (Dwork and Roth, 2014, Theorem 3.13).

Replacing the Laplace distribution with the one-sided Laplace distribution makes the privacy proof only a bit more difficult. While privacy still does not follow from the standard composition theorem—in fact, there is now nothing to compose because sampling from the one-sided Laplace distribution isn't differentially private—we can give a similar proof by approximate coupling. The main difference is in the rule [LAPGEN]: the analogous rule for the one-sided Laplace distribution has a slightly stronger pre-condition, with $0 \leq k + e_1\langle 1\rangle - e_2\langle 2\rangle \leq k'$ in place of $|k + e_1\langle 1\rangle - e_2\langle 2\rangle| \leq k'$. Our coupling proof is otherwise unchanged.

```
i ← 1; out ← [];
t ⟵$ Lap_{ε/4}(T);
while i ≤ N ∧ |out| < C do
    ans ← (0, 0); go ← true;
    while i ≤ N ∧ go do
        a ⟵$ Lap_{ε/8C}(evalQ(i, d));
        if a > t then
            noisy ⟵$ Lap_{ε/4C}(evalQ(i, d));
            ans ← (i, noisy);
            out ← ans :: out;
            go ← false;
    i ← i + 1
```

Figure 4.6: Sparse Vector

The Sparse Vector mechanism

Our second example is the *Sparse Vector mechanism*, a well-known algorithm with a challenging privacy proof. At least six variants were thought to be proved private, only for subtle mistakes to later surface in four of them (Lyu et al., 2017). Perhaps the canonical (correct) version of the algorithm can be found in the textbook by Dwork and Roth (2014), where it is called NUMERICSPARSE. This mechanism takes a numeric *threshold* $T \in \mathbb{Z}$, a cutoff $C \in \mathbb{N}$, a list of numeric queries $q_1, \ldots, q_N : \mathcal{D} \to \mathbb{Z}$, and a database $d \in \mathcal{D}$ as input. Sparse Vector releases the indices of the first C queries that have answer approximately above the threshold, along with noisy answers for each of these queries. The mechanism adds Laplace noise to the threshold and Laplace noise to each query answer, returning the queries with noisy answer above the noisy threshold. Again, the challenge in the privacy analysis is to only pay for above-threshold queries, rather than all queries.[4]

Figure 4.6 presents the code for the Sparse Vector algorithm. The variable *out* stores a list of pairs of an index and a noisy answer for each query that is approximately above-threshold; the list is initially empty and pairs are added to the head. The algorithm stops when it answers C queries or when it processes all N queries. The code is structured in a slightly artificial way—the queries are broken into chunks, where each iteration of the outer loop corresponds to exactly one above-threshold query. In their presentation, Dwork and Roth (2014) first prove privacy for a subroutine called ABOVETHRESHOLD—which randomizes the threshold and executes one iteration of the outer loop—by carefully manipulating conditional probabilities. They then verify the whole mechanism NUMERICSPARSE by composing calls to ABOVETHRESHOLD and applying the standard composition theorem (Theorem 4.1.4).

Rather than re-randomize the threshold after every answered query, we add noise to the threshold just at the beginning of the algorithm; this variant was independently proposed by Lyu et al. (2017). Accordingly, it is no longer possible to analyze the outer loop iterations via standard privacy composition since each iteration of the outer loop is not differentially private. While using the same noise for the threshold does not affect the asymptotic accuracy of Sparse Vector, practical applications may benefit.

Theorem 4.7.5. *Suppose each query q_i is 1-sensitive: $|q_i(d) - q_i(d')| \leq 1$ for adjacent databases d, d', and the threshold T is the same for both runs. Then Sparse Vector is ε-differentially private.*

Proof by approximate coupling. We first couple the threshold sampling so $t\langle 1 \rangle + 1 = t\langle 2 \rangle$. The means are 0 apart, the coupled samples are 1 apart, and the noise is from the Laplace distribution with parameter $\varepsilon/4$, so this is an $(1 \cdot \varepsilon/4, 0) = (\varepsilon/4, 0)$ approximate coupling. Assuming this coupling, we argue that the two executions of the

---

[4]A precursor of this algorithm was designed to release a private version of a vector of numbers where most of the entries are known to be zero, i.e., a sparse vector.

inner loop can be approximately coupled to satisfy $ans\langle 1 \rangle = ans\langle 2 \rangle$. We consider the inner loop and construct an approximate coupling such that if $ans\langle 1 \rangle = (j, v)$ then $ans\langle 2 \rangle = (j, v)$ as well.

Just like we did in the proof of Report-noisy-max, we use different couplings depending on where we are in the loop relative to iteration j. In iterations before j, we use the null coupling when sampling a and *noisy* to give a $(0, 0)$-approximate coupling such that $|a\langle 1 \rangle - a\langle 2 \rangle| \leq 1$; this ensures that if we don't go above threshold in the first execution before j, then we also don't go above threshold in the second execution before j. We take the same $(0, 0)$-coupling for iterations after j. In the critical iteration j, we couple the samplings for a to ensure $a\langle 1 \rangle + 1 = a\langle 2 \rangle$ and we couple $noisy\langle 1 \rangle = noisy\langle 2 \rangle$ if necessary. Combined with the threshold coupling $t\langle 1 \rangle + 1 = t\langle 2 \rangle$, this ensures that if we go above threshold in iteration j in the first execution then we also go above threshold in iteration j in the second execution, and the noisy answers for above-threshold queries are equal.

To compute the approximation parameters, the coupling for a is an $(\varepsilon/4C, 0)$-approximate coupling: the distance between the coupled samples is at most 2 greater than the distance between the means, and the noise is drawn from $\mathbf{Lap}_{\varepsilon/8C}$. The coupling for *noisy* is the standard coupling for the Laplace mechanism; it is an $(\varepsilon/4C, 0)$-approximate coupling since the true answers are at most 1 apart and the noise is drawn from $\mathbf{Lap}_{\varepsilon/4C}$. So, iteration j uses an $(\varepsilon/4C + \varepsilon/4C, 0) = (\varepsilon/2C, 0)$-approximate coupling and the loop coupling ensures that if $ans\langle 1 \rangle = (j, v)$ then $ans\langle 2 \rangle = (j, v)$. This gives an $(\varepsilon/2C, 0)$-approximate coupling for the inner loop ensuring $ans\langle 1 \rangle = ans\langle 2 \rangle$ and preserving $out\langle 1 \rangle = out\langle 2 \rangle$.

Since there are at most C iterations of the outer loop, we have an $(\varepsilon/2, 0)$-approximate coupling ensuring $out\langle 1 \rangle = out\langle 2 \rangle$ at the end of the algorithm. Combined with the $(\varepsilon/2, 0)$-coupling for the threshold, this shows that Sparse Vector is $(\varepsilon, 0)$-differentially private. $\qquad\square$

Remark 4.7.6. Earlier versions of Sparse Vector returned the noisy answers without adding fresh noise. These variants are now known to be incorrect: Lyu et al. (2017) show they are not ε-differentially private for any finite ε. If we attempt a proof by approximate coupling we immediately hit a problem: after coupling $a\langle 1 \rangle + 1 = a\langle 2 \rangle$, returning the noisy answer a is not differentially private. By itself, this obstacle doesn't show the algorithm is not differentially private. However, it suggests that something may be amiss.

We can also give a more formal proof of privacy in APRHL. Like we did for Report-noisy-max, we transform the loops in order to apply couplings with different costs in different iterations. Sparse Vector also introduces an additional complication: under the we will build coupling, we don't know the inner loops remain synchronized. So, we work with the following, equivalent implementation:

$$
\begin{aligned}
&i \leftarrow 1; out \leftarrow [\,]; \\
&t \xleftarrow{\$} \mathbf{Lap}_{\varepsilon/4}(T); \\
&\textbf{while } i \leq N \wedge |out| < C \textbf{ do} \\
&\quad ans \leftarrow (0, 0); go \leftarrow \textbf{true}; \\
&\quad \textbf{while } i \leq N \textbf{ do} \\
&\quad\quad a \xleftarrow{\$} \mathbf{Lap}_{\varepsilon/8C}(\textbf{evalQ}(i, d)); \\
&\quad\quad \textbf{if } a > t \wedge go \textbf{ then} \\
&\quad\quad\quad noisy \xleftarrow{\$} \mathbf{Lap}_{\varepsilon/4C}(\textbf{evalQ}(i, d)); \\
&\quad\quad\quad ans \leftarrow (i, noisy); \\
&\quad\quad\quad go \leftarrow \textbf{false}; \\
&\quad\quad i \leftarrow i + 1; \\
&\quad \textbf{if } p_1(ans) \neq 0 \textbf{ then} \\
&\quad\quad out \leftarrow ans :: out; i \leftarrow p_1(ans) + 1
\end{aligned}
$$

Compared to the more straightforward implementation in Fig. 4.6, the main difference is that the inner loop passes over all the queries. Once the inner loop finds an above threshold query, the algorithm sets the flag go and the inner loop skips over all remaining queries. Then if an above-threshold query was found in the inner loop, the index in ans must be non-zero. In this case, the outer loop records the answer and resets the counter to the query

after the most recent above-threshold query (recall p_1 returns the first element of a pair). By running through all the queries, the inner loops can be analyzed in a synchronized fashion. We call the inner loop *aboveT*, and the whole program program *sparseV*.

Theorem 4.7.7. *Suppose each query q_i is 1-sensitive: $|q_i(d) - q_i(d')| \leq 1$ for adjacent databases d, d' and the threshold T is the same for both runs. Then the following judgment is derivable in* APRHL*:*

$$\vdash sparseV \sim_{(\varepsilon, 0)} sparseV : Adj(d\langle 1\rangle, d\langle 2\rangle) \implies out\langle 1\rangle = out\langle 2\rangle$$

Proof. We elide the adjacency assertion $Adj(d\langle 1\rangle, d\langle 2\rangle)$ and synchronization assertion $i\langle 1\rangle = i\langle 2\rangle$ since they are preserved throughout the proof. Let's first consider the inner loop *aboveT*. We prove the following judgment for every pair $(j, v) \in \mathbb{N} \times \mathbb{Z}$:

$$\vdash aboveT \sim_{(\varepsilon/2C, 0)} aboveT : t\langle 1\rangle + 1 = t\langle 2\rangle \implies ans\langle 1\rangle = (j, v) \rightarrow ans\langle 2\rangle = (j, v)$$

The cases $j = 0$ and $j > N$ are trivial, so we consider $j \in [1, N]$. Much like we did for Report-noisy-max, we split the loop into three pieces: iterations before j, iteration j, and iterations after j.

> **while** $i \leq N \wedge i < j$ **do**
> $a \xleftarrow{\$} \mathbf{Lap}_{\varepsilon/8C}(\mathbf{evalQ}(i, d))$;
>   **if** $a > t \wedge go$ **then**
> $noisy \xleftarrow{\$} \mathbf{Lap}_{\varepsilon/4C}(\mathbf{evalQ}(i, d))$;
> $ans \leftarrow (i, noisy); go \leftarrow \mathbf{false}$;
> $i \leftarrow i + 1$;
> **while** $i \leq N \wedge i = j$ **do**
> $a \xleftarrow{\$} \mathbf{Lap}_{\varepsilon/8C}(\mathbf{evalQ}(i, d))$;
>   **if** $a > t \wedge go$ **then**
> $noisy \xleftarrow{\$} \mathbf{Lap}_{\varepsilon/4C}(\mathbf{evalQ}(i, d))$;
> $ans \leftarrow (i, noisy); go \leftarrow \mathbf{false}$;
> $i \leftarrow i + 1$;
> **while** $i \leq N$ **do**
> $a \xleftarrow{\$} \mathbf{Lap}_{\varepsilon/8C}(\mathbf{evalQ}(i, d))$;
>   **if** $a > t \wedge go$ **then**
> $noisy \xleftarrow{\$} \mathbf{Lap}_{\varepsilon/4C}(\mathbf{evalQ}(i, d))$;
> $ans \leftarrow (i, noisy); go \leftarrow \mathbf{false}$;
> $i \leftarrow i + 1$

We call this program *aboveT'*, the loops $w_<$, $w_=$, and $w_>$, and body of the loop *body*. We take invariants:

$$\Theta_< \triangleq \begin{cases} t\langle 1\rangle + 1 = t\langle 2\rangle \\ go\langle 1\rangle \rightarrow go\langle 2\rangle \\ \neg(i\langle 1\rangle \leq N \wedge i\langle 1\rangle < j) \rightarrow i\langle 1\rangle = j \end{cases}$$

$$\Theta_= \triangleq \begin{cases} t\langle 1\rangle + 1 = t\langle 2\rangle \\ go\langle 1\rangle \rightarrow go\langle 2\rangle \\ ans\langle 1\rangle = (j, v) \rightarrow ans\langle 2\rangle = (j, v) \\ \neg(i\langle 1\rangle \leq N \wedge i\langle 1\rangle = j) \rightarrow i\langle 1\rangle = j + 1 \end{cases}$$

$$\Theta_> \triangleq \begin{cases} t\langle 1\rangle + 1 = t\langle 2\rangle \\ i\langle 1\rangle > j \\ ans\langle 1\rangle = (j, v) \rightarrow ans\langle 2\rangle = (j, v) \end{cases}$$

We begin with the first loop. To show

$$\vdash body \sim_{(0,0)} body : i\langle 1 \rangle \leq N \wedge i\langle 1 \rangle < j \wedge \Theta_< \implies \Theta_<,$$

we couple the sampling for a with the null coupling $[\text{LapNull}]$ so that

$$|a\langle 1 \rangle - a\langle 2 \rangle| = |\textbf{evalQ}(i\langle 1 \rangle, d\langle 1 \rangle) - \textbf{evalQ}(i\langle 2 \rangle, d, \langle 2 \rangle)| \leq 1.$$

For the conditional statements we use the one-sided rules $[\text{Cond-L}]$ and $[\text{Cond-R}]$, giving four possible cases for the guard $a > t \wedge go$ in the two executions:

(True, True) We use $[\text{LapNull}]$ to couple the samplings for *noisy* and establish $\neg go\langle 1 \rangle$.

(True, False) We use $[\text{Lap-L}]$ for sampling $noisy\langle 1 \rangle$ to establish $\neg go\langle 1 \rangle$.

(False, True) If $go\langle 1 \rangle$ is false, then we use $[\text{Lap-R}]$ for sampling $noisy\langle 2 \rangle$ and conclude $go\langle 1 \rangle \to go\langle 2 \rangle$.

If $go\langle 1 \rangle$ is true, then $a\langle 1 \rangle$ must be below threshold but this case is impossible: $a\langle 2 \rangle$ must be above threshold but the thresholds are coupled so that $t\langle 1 \rangle + 1 = t\langle 2 \rangle$ and $|a\langle 1 \rangle - a\langle 2 \rangle| \leq 1$.

(False, False) We use $[\text{Skip}]$, preserving $go\langle 1 \rangle \to go\langle 2 \rangle$.

Putting the cases together, we have

$$\vdash body \sim_{(0,0)} body : \Theta_< \implies \Theta_<.$$

Since the loops are synchronized we apply $[\text{While}]$ to get

$$\vdash w_< \sim_{(0,0)} w_< : \Theta_< \implies \Theta_< \wedge \neg(i\langle 1 \rangle \leq N \wedge i\langle 1 \rangle < j).$$

Next, we turn to the second loop. We couple the samplings for a so that

$$a\langle 1 \rangle + 1 = a\langle 2 \rangle$$

with $[\text{LapGen}]$, taking $k = 1$, $k' = 2$. Since the parameter for the Laplace sampling is $\varepsilon/8C$, this is a $(2 \cdot \varepsilon/8C, 0) = (\varepsilon/4C, 0)$-approximate coupling. Like for the first loop, we have four cases when analyzing the conditional. The most interesting case is when both guards are true, when we couple the samplings for *noisy* with the standard Laplace rule $[\text{Lap}]$ so that $noisy\langle 1 \rangle = noisy\langle 2 \rangle$; this is an $(\varepsilon/4C, 0)$-approximate coupling since the queries are 1-sensitive. We wind up with $\neg go\langle 1 \rangle$ and $\neg go\langle 2 \rangle$, establishing the post-condition $go\langle 1 \rangle \to go\langle 2 \rangle$. Moreover,

$$ans\langle 1 \rangle = (j, v) \to ans\langle 2 \rangle = (j, v)$$

under the coupling. This suffices to establish the invariant $\Theta_=$ when both guards are true. We use a similar argument for the other three cases, proving

$$\vdash body \sim_{(\varepsilon/2C,0)} body : \Theta_= \implies \Theta_=.$$

Since there is exactly one iteration, $[\text{While}]$ gives

$$\vdash w_= \sim_{(\varepsilon/2C,0)} w_= : \Theta_= \implies \Theta_= \wedge \neg(i\langle 1 \rangle \leq N \wedge i\langle 1 \rangle = j).$$

In the last loop, we couple the samplings for a with $[\text{LapNull}]$ and the samplings for *noisy* with $[\text{LapNull}]$ or the one-sided rules $[\text{Lap-L}]$ or $[\text{Lap-R}]$, depending on whether the guards are true or not. This gives

$$\vdash w_> \sim_{(0,0)} w_> : \Theta_> \implies \Theta_> \wedge \neg(i\langle 1 \rangle \leq N).$$

After using the rule of consequence with implications

$$\models \Theta_< \wedge \neg(i\langle 1\rangle \leq N \wedge i\langle 1\rangle < j) \rightarrow \Theta_=$$
$$\models \Theta_= \wedge \neg(i\langle 1\rangle \leq N \wedge i\langle 1\rangle = j) \rightarrow \Theta_>,$$

we apply [SEQ] to combine the loop judgments and sum the approximation parameters:

$$\vdash aboveT' \sim_{(\varepsilon/2C,0)} aboveT' : t\langle 1\rangle + 1 = t\langle 2\rangle \Longrightarrow ans\langle 1\rangle = (j,v) \rightarrow ans\langle 2\rangle = (j,v).$$

By applying pointwise equality [PW-EQ] and then the frame rule [FRAME] to preserve the threshold coupling, we establish the desired judgment for the inner loop:

$$\vdash aboveT' \sim_{(\varepsilon/2C,0)} aboveT' : t\langle 1\rangle + 1 = t\langle 2\rangle \Longrightarrow ans\langle 1\rangle = ans\langle 2\rangle \wedge t\langle 1\rangle + 1 = t\langle 2\rangle.$$

Now we turn to the outer loop w of $sparseV$. At the end of each iteration, we know

$$i\langle 1\rangle = i\langle 2\rangle \wedge out\langle 1\rangle = out\langle 2\rangle \wedge t\langle 1\rangle + 1 = t\langle 2\rangle$$

since the inner loop guarantees $ans\langle 1\rangle = ans\langle 2\rangle$. Applying [WHILE] with decreasing variant

$$e_v \triangleq (i = N)\,?\,0 : C - |out|,$$

there at most C iterations and each iteration is related by an $(\varepsilon/2C,0)$-coupling. So we have the following judgment for the outer loop:

$$\vdash w \sim_{(\varepsilon/2,0)} w : out\langle 1\rangle = out\langle 2\rangle \wedge t\langle 1\rangle + 1 = t\langle 2\rangle \Longrightarrow out\langle 1\rangle = out\langle 2\rangle.$$

Finally, we ensure the loop pre-condition $t\langle 1\rangle + 1 = t\langle 2\rangle$ by coupling the sampling instructions for t with [LAPGEN], taking $k, k' \triangleq 1$. Since the Laplace distribution has parameter $\varepsilon/2$, this is an $(\varepsilon/2,0)$-approximate coupling. Putting everything together we have

$$\vdash sparseV \sim_{(\varepsilon,0)} sparseV : Adj(d\langle 1\rangle, d\langle 2\rangle) \Longrightarrow out\langle 1\rangle = out\langle 2\rangle,$$

showing that Sparse Vector is ε-differentially private. $\square$

Remark 4.7.8. It would be a bit more natural to use the guard $go = \textbf{false}$ in the final conditional, but showing $go\langle 1\rangle = go\langle 2\rangle$ after the inner loop is not so easy: our proof can only establish $go\langle 1\rangle \rightarrow go\langle 2\rangle$. In order to verify the program with guard $go = \textbf{false}$, we would need the one-sided invariant

$$p_1(ans) \neq 0 \leftrightarrow go = \textbf{false}$$

on both sides. While this invariant does hold, here we hit a limitation of the pointwise equality rule [PW-EQ]: the post-condition is narrowly restricted and we cannot show the above invariant in the post-condition of the inner loop. Later in Chapter 5 we will see how to leverage these one-sided invariants (cf. rules [AND-L] and [AND-R]).

4.8   Discussion

To close this chapter, we briefly survey related systems for formally verifying differential privacy and discuss other applications of approximate couplings.

Formal verification of differential privacy

Due to its rich composition properties and compelling motivations, differential privacy is an attractive target for formal verification. Researchers have considered a broad array of techniques including linear types (Azevedo de Amorim, Gaboardi, Gallego Arias, and Hsu, 2014; Azevedo de Amorim, Gaboardi, Hsu, Katsumata, and Cherigui, 2017; Reed and Pierce, 2010; Winograd-Cort, Haeberlen, Roth, and Pierce, 2017), sized types (Gaboardi, Haeberlen, Hsu, Narayan, and Pierce, 2013), product programs (Barthe et al., 2014b), refinement types (Barthe, Gaboardi, Gallego Arias, Hsu, Roth, and Strub, 2015b), and more (Ebadi, Antignac, and Sands, 2016; Ebadi and Sands, 2016; Ebadi, Sands, and Schneider, 2015; McSherry, 2009; Palamidessi and Stronati, 2012; Proserpio, Goldberg, and McSherry, 2014; Tschantz, Kaynar, and Datta, 2011). (Readers can consult the recent survey by Barthe, Gaboardi, Hsu, and Pierce (2016d) for a more comprehensive overview.)

Most existing techniques cannot verify privacy proofs beyond composition, such as the two examples we presented in this chapter. One notable exception is the LIGHTDP system proposed by Zhang and Kifer (2017), which combines a relational, dependent type system with a product program construction. This system can prove privacy for the Sparse Vector mechanism with a high degree of automation by using a novel type inference algorithm and a MAXSAT solver to optimize the privacy cost.

The key theoretical idea behind LIGHTDP is *randomness alignment*, which specifies an injection from one sample space to another while recording the difference in probabilities. Randomness alignments are similar to the approximate couplings we saw for the Laplace mechanism (e.g., in the rules [LAPNULL] and [LAPGEN]). One important novelty in LIGHTDP is that alignments can be selected lazily, based on the result of the sample in the first execution. In this way, LIGHTDP can sometimes construct a privacy proof in one shot where APRHL would need to reason about each output separately with [PW-EQ]. In the Sparse Vector mechanism, for instance, LIGHTDP can select the shift coupling when the first iteration goes above threshold, and use the null coupling when it does not. (This approach does not work for Report-noisy-max, as the iteration with the highest noisy score is not known until the program has finished executing.) This lazy choice of alignment can be modeled by an approximate coupling that selects between two couplings, depending on a predicate on the first sample. If the predicate and two couplings satisfy a technical non-overlapping condition, the result is again an approximate coupling.

Theorem 4.8.1 (Choice coupling). *Let μ_1, μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$. Suppose we have a predicate $\mathcal{P} \subseteq \mathcal{A}_1$ and two approximate couplings*

$$\mu_1 \, \mathcal{R}^{\sharp(\varepsilon,\delta)} \, \mu_2 \quad and \quad \mu_1 \, \mathcal{S}^{\sharp(\varepsilon,\delta)} \, \mu_2$$

such that the following non-overlapping *condition holds:*

$$\mathcal{R}(\mathcal{P}) \cap \mathcal{S}(\mathcal{A}_1 \setminus \mathcal{P}) = \varnothing,$$

where $\mathcal{R}(\mathcal{P})$ is the set of elements in $\mathcal{A}_2$ related to something in $\mathcal{P}$ under $\mathcal{R}$, and $\mathcal{S}(\mathcal{A}_1 \setminus \mathcal{P})$ is the set of elements in $\mathcal{A}_2$ related to something outside of $\mathcal{P}$ under $\mathcal{S}$. Then there is an approximate coupling

$$\mu_1 \, \mathcal{T}^{\sharp(\varepsilon,2\delta)} \, \mu_2$$

where $\mathcal{T}$ is the relation

$$\mathcal{T} \triangleq \{(a_1, a_2) \mid (a_1 \in \mathcal{P} \rightarrow (a_1, a_2) \in \mathcal{R}) \wedge (a_1 \notin \mathcal{P} \rightarrow (a_1, a_2) \in \mathcal{S})\}.$$

Proof. Let ρ_L, ρ_R and σ_L, σ_R witness the two approximate couplings. Define witnesses

$$\mu_L(a_1, a_2) \triangleq \begin{cases} \rho_L(a_1, a_2) & : a_1 \in \mathcal{P} \\ \sigma_L(a_1, a_2) & : a_1 \notin \mathcal{P} \\ 0 & : a_1 = \star \end{cases} \quad and \quad \mu_R(a_1, a_2) \triangleq \begin{cases} \rho_R(a_1, a_2) & : a_1 \in \mathcal{P} \\ \sigma_R(a_1, a_2) & : a_1 \notin \mathcal{P} \\ \mu_2(a_2) - \sum_{a_1' \in \mathcal{A}_1} \mu_R(a_1', a_2) & : a_1 = \star. \end{cases}$$

The support and marginal conditions are immediate. The main thing to show is that $\mu_R(\star, a_2)$ is non-negative; it suffices to show $\sum_{a_1' \in \mathcal{A}_1} \mu_R(a_1', a_2) \leq \mu_2(a_2)$. There are three cases: either $a_2 \in \mathcal{R}(\mathcal{P})$, $a_2 \in \mathcal{S}(\mathcal{A}_1 \setminus \mathcal{P})$, or none of the above; by the non-overlapping condition, these cases are mutually exclusive. In the first case, we have

$$\sum_{a_1' \in \mathcal{A}_1} \mu_R(a_1', a_2) = \sum_{a_1' \in \mathcal{P}} \rho_R(a_1', a_2) + \sum_{a_1' \in \mathcal{A} \setminus \mathcal{P}} \sigma_R(a_1', a_2) = \sum_{a_1' \in \mathcal{P}} \rho_R(a_1', a_2) \leq \mu_2(a_2).$$

The second case is similar:

$$\sum_{a_1' \in \mathcal{A}_1} \mu_R(a_1', a_2) = \sum_{a_1' \in \mathcal{P}} \rho_R(a_1', a_2) + \sum_{a_1' \in \mathcal{A} \setminus \mathcal{P}} \sigma_R(a_1', a_2) = \sum_{a_1' \in \mathcal{A} \setminus \mathcal{P}} \sigma_R(a_1', a_2) \leq \mu_2(a_2).$$

In the third case the inequality clearly holds, as the sum is equal to 0.

It only remains to check the distance condition $d_\varepsilon(\mu_L, \mu_R) \leq 2\delta$. By the distance conditions on the given witnesses, there are non-negative constants $\zeta(a_1, a_2), \xi(a_1, a_2)$ such that

$$\rho_L(a_1, a_2) \leq \exp(\varepsilon) \cdot \rho_R(a_1, a_2) + \zeta(a_1, a_2) \quad \text{and} \quad \sigma_L(a_1, a_2) \leq \exp(\varepsilon) \cdot \sigma_R(a_1, a_2) + \xi(a_1, a_2)$$

with bounded sums:

$$\sum_{a_1, a_2} \zeta(a_1, a_2) \leq \delta \qquad \text{and} \qquad \sum_{a_1, a_2} \xi(a_1, a_2) \leq \delta.$$

By definition, we have

$$\mu_L(a_1, a_2) \leq \exp(\varepsilon) \cdot \mu_R(a_1, a_2) + \max(\zeta(a_1, a_2), \xi(a_1, a_2))$$

for all $a_1, a_2 \neq \star$; it is easy to check

$$\mu_L(a_1, a_2) \leq \exp(\varepsilon) \cdot \mu_R(a_1, a_2)$$

when $a_1 = \star$ or $a_2 = \star$. We can bound the sums

$$\sum_{a_1, a_2} \max(\zeta(a_1, a_2), \xi(a_1, a_2)) \leq \sum_{a_1, a_2} \zeta(a_1, a_2) + \xi(a_1, a_2) \leq 2\delta$$

to give the claimed distance condition. Thus μ_L, μ_R witness the desired approximate coupling. □

However, this coupling is not quite precise enough: its cost is greater than the *maximum* cost of the two couplings. Taking the example of Sparse Vector again, the shift coupling [LAPGEN] has a non-zero cost while the null coupling [LAPNULL] has zero cost. If we are selecting between these two couplings, we do not want to pay for the (more expensive) [LAPGEN] coupling on every iteration, but only on the single iteration where the first execution goes above threshold.

LIGHTDP features a more fine-grained analysis where the cost can depend on which choice was taken. Since the choice depends on whether the first sample satisfies a predicate (e.g., goes above threshold), this analysis involves a *randomized* notion of privacy cost; LIGHTDP uses a product construction as a secondary analysis to bound the parameters in all possible executions. In contrast, APRHL requires the approximation parameters to be constant at each stage, though a more general form of approximate coupling allowing variable costs for different samples enables LIGHTDP-style privacy proofs. (See Chapter 6 for further discussion.)

Approximate couplings in formal verification

Approximate liftings are a flexible abstraction for reasoning about differential privacy. While we have focused on program logics, approximate liftings have played a central role in other formal verification settings.

Barthe et al. (2014b) show how to verify differential privacy by first interleaving two programs into a single program and then analyzing the result, a so-called "synchronized product" approach. Their construction replaces

every pair of corresponding random sampling commands with a single, non-deterministic assignment of a pair, along with a specification of the relation between the returned values. In this way, they can verify differential privacy by constructing proofs in non-deterministic Hoare logic. Their technique is based on approximate liftings and roughly corresponds to the fragment of APRHL where all conditionals are synchronized under the coupling, so only pairs of identical programs are related.

Approximate liftings can also play a useful role in type systems. Barthe et al. (2015b) propose a relational refinement type system for a functional language HOARE^2. To handle relational reasoning for distributions, their system features a probability monad over a relation $\mathcal{R}$ on the base type, indexed by approximation parameters. This monad is then interpreted as an approximate lifting with support contained in $\mathcal{R}$. In their typing rule for monadic bind with initial distributions related by a $\mathcal{R}$-lifting, the body is typed under the assumption that the samples are related by $\mathcal{R}$, giving a clean way to use information about distributions when reasoning about samples. This principle can be seen in the APRHL rule [SEQ] or more abstractly, as a monadic composition principle for approximate liftings.

Barthe et al. (2015b) also explore an interesting application of approximate liftings: given sub-distributions μ_1, μ_2 over the unit interval $[0, 1]$,[5] the approximate lifting

$$\mu_1 \left(\leq\right)^{\sharp(\varepsilon, \delta)} \mu_2$$

implies a bound on expected values: $\mathbb{E}_{x_1 \sim \mu_1}[x_1] \leq \exp(\varepsilon) \cdot \mathbb{E}_{x_2 \sim \mu_2}[x_2] + \delta$; this can be seen as a consequence of approximate stochastic domination. Barthe et al. (2015b) use this observation to prove relational properties involving expectations for algorithms at the intersection of mechanism design and differential privacy, where the mechanisms are randomized and the incentive properties follow from differential privacy. Barthe, Gaboardi, Gallego Arias, Hsu, Roth, and Strub (2016b) use similar ideas to verify more sophisticated incentive properties.

---

[5]More precisely, a discrete version of the unit interval $[0, 1]$.

Chapter 5

Advanced approximate couplings

In the previous chapter, we saw how approximate couplings of the Laplace distribution and the pointwise equality principle support new proofs of privacy by approximate coupling. To enhance the power of this proof technique, we develop the theory of approximate couplings further in this chapter, giving a potpourri of new constructions and showing equivalences with other notions of approximate lifting. Our results enable richer proofs by approximate coupling, capable of modeling more advanced proofs of privacy.

To begin, we show that approximate couplings are a discrete version of the approximate lifting recently proposed by Sato (2016). This equivalence gives a highly convenient method for constructing approximate couplings and extends a classical result by Strassen (1965) for probabilistic couplings (Section 5.1). Then, we consider two new constructions: *up-to-bad* approximate coupling (Section 5.2) and *optimal subset* coupling (Section 5.3). To follow, we identify a symmetric version of approximate coupling that supports an advanced composition principle generalizing the advanced composition theorem of differential privacy (Section 5.4). To make our constructions concrete, we introduce new APRHL proof rules and prove differential privacy for the *Between Thresholds* mechanism, recently proposed by Bun et al. (2017) (Section 5.5). Finally, we show approximate couplings unify several previously proposed notions (Section 5.6). Taken together, our equivalences and constructions serve as strong evidence that we have arrived at a natural, approximate generalization of probabilistic coupling.

5.1 Equivalence with Sato's approximate lifting

So far, we have considered approximate couplings for discrete distributions. In recent work, Sato (2016) develops a version of APRHL where programs can sample from continuous distributions, like the Laplace and Gaussian distributions. Intriguingly, Sato takes a significantly different definition of approximate lifting as the foundation of his logic. In the discrete case, his definition is as follows.

Definition 5.1.1 (Sato (2016))**.** Let μ_1 and μ_2 be sub-distributions over countable sets $\mathcal{A}_1$ and $\mathcal{A}_2$, and let $\mathcal{R} \subseteq \mathcal{A}_1 \times \mathcal{A}_2$ be a relation. There is an (ε, δ)-*approximate $\mathcal{R}$-lifting* of (μ_1, μ_2) if for every subset $\mathcal{S}_1 \subseteq \mathcal{A}_1$, the following inequality holds:

$$\mu_1(\mathcal{S}_1) \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{R}(\mathcal{S}_1)) + \delta.$$

(Recall $\mathcal{R}(\mathcal{S}_1)$ is the subset of $\mathcal{A}_2$ that is related to some element in $\mathcal{S}_1$ under $\mathcal{R}$.)

This definition is interesting for several reasons. First, rather than requiring the existence of witness distributions, Sato's definition quantifies over all subsets of samples. Second, Sato shows that his definition generalizes the prior definition of approximate lifting by Barthe and Olmedo (2013) and Olmedo (2014), leaving open the question of whether they are equivalent; in fact, they are not. However, we show our definition of approximate lifting (Definition 4.2.2) is equivalent to Sato's definition in the discrete case. Our result can be seen as an approximate version of Strassen's theorem (Theorem 2.1.11); it also implies Strassen's theorem for discrete sub-distributions.

One direction of the equivalence is not hard to show.

Theorem 5.1.2 (Approximate lifting implies Sato's lifting)**.** *Let μ_1 and μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$, and let $\mathcal{R} \subseteq \mathcal{A}_1 \times \mathcal{A}_2$ be a binary relation. Suppose there exists an approximate lifting*

$$\mu_1 \, \mathcal{R}^{\sharp(\varepsilon,\delta)} \, \mu_2.$$

Then $\mu_1(\mathcal{S}_1) \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{R}(\mathcal{S}_1)) + \delta$ for every subset $\mathcal{S}_1 \subseteq \mathcal{A}_1$.

Proof. Let μ_L, μ_R witness the approximate lifting. By the distance, support, and marginal conditions,

$$\begin{aligned}
\mu_1(\mathcal{S}_1) &= \mu_L(\mathcal{S}_1 \times \mathcal{A}_2^\star) \\
&\leq \exp(\varepsilon) \cdot \mu_R(\mathcal{S}_1 \times \mathcal{A}_2^\star) + \delta \\
&= \exp(\varepsilon) \cdot \mu_R(\mathcal{S}_1 \times \mathcal{R}(\mathcal{S}_1)) + \delta \\
&\leq \exp(\varepsilon) \cdot \mu_R(\mathcal{A}_1^\star \times \mathcal{R}(\mathcal{S}_1)) + \delta \\
&= \exp(\varepsilon) \cdot \mu_R(\mathcal{R}(\mathcal{S}_1)) + \delta. \qquad \square
\end{aligned}$$

The other direction—showing Sato's approximate lifting implies our approximate lifting—is a bit more involved. We proceed in two steps. First, we prove the implication for distributions over finite sets. Then we generalize to distributions over countable sets by a limiting argument.

The finite case

The finite case follows from the max flow-min cut theorem. Roughly speaking, Sato's condition ensures that in a certain graph, the minimum cut is not too small so the maximum flow must be large. This will imply we can build witnesses to the approximate lifting from the maximum flow. First, we recall the classical max flow-min cut theorem (see any standard textbook on algorithms, e.g., Kleinberg and Tardos (2005)).

Theorem 5.1.3 (Max flow-min cut)**.** *Let G be a finite graph with vertices V and directed edges E. Let $s \in V$ be the* source *node (i.e., there are no edges $(a,s) \in E$) and let $t \in V$ be the* sink *node (i.e., there are no edges $(t,b) \in E$); we assume s and t are unique. We suppose each edge has* capacity *$c(a,b) \in \mathbb{R} \cup \{\infty\}$. A* flow *from s to t is a map $f : E \to \mathbb{R}^+$ such that (i) the flow is conserved at each internal node:*

$$\sum_{(a,v) \in E} f(a,v) = \sum_{(v,b) \in E} f(v,b)$$

for every node $v \neq s, t$, and (ii) the flow respects the capacity constraints: $f(a,b) \leq c(a,b)$. The weight *of a flow $|f|$ is the amount of flow leaving s; by conservation, this is equal to the total flow entering t. A* cut *C is a partition of the vertices into two sets (V_1, V_2). The* capacity *of a cut $|C|$ is the total capacity of all edges (a,b) crossing (V_1, V_2), i.e., with $a \in V_1$ and $b \in V_2$.*

The weight of the largest flow equals the minimum capacity of a cut (V_1, V_2) with $s \in V_1$ and $t \in V_2$.

Theorem 5.1.4. *Let μ_1 and μ_2 be sub-distributions with finite support over sets $\mathcal{A}_1$ and $\mathcal{A}_2$, and let $\mathcal{R} \subseteq \mathcal{A}_1 \times \mathcal{A}_2$ be a binary relation such that $\mu_1(\mathcal{S}_1) \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{R}(\mathcal{S}_1)) + \delta$ for every $\mathcal{S}_1 \subseteq \mathcal{A}_1$. Then there exists an approximate lifting*

$$\mu_1 \, \mathcal{R}^{\sharp(\varepsilon,\delta)} \, \mu_2.$$

Proof. Without loss of generality, by Theorem 4.2.7 we may take $\mathcal{A}_1$ and $\mathcal{A}_2$ to be the supports of μ_1 and μ_2 respectively; these are finite by assumption. We define a finite graph with vertices $\mathcal{A}_1^\star \cup \mathcal{A}_2^\star \cup \{\top, \bot\}$. Note that we

take two distinct vertices $\star_1, \star_2$ corresponding to the $\star$ elements in $\mathcal{A}_1^\star$ and $\mathcal{A}_2^\star$. We connect the source $\top$ to every element of $\mathcal{A}_1^\star$ with capacities

$$c(\top, a_1) \triangleq \mu_1(a_1) \cdot \exp(-\varepsilon)$$
$$c(\top, \star_1) \triangleq \omega - \exp(-\varepsilon) \cdot \mu_1(\mathcal{A}_1),$$

where $\omega \triangleq \mu_2(\mathcal{A}_2) + \exp(-\varepsilon) \cdot \delta$. Now $c(\top, \star_1) \geq 0$ since by assumption,

$$\mu_1(\mathcal{A}_1) \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{R}(\mathcal{A}_1)) + \delta \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{A}_2).$$

We connect every element of $\mathcal{A}_2^\star$ to the sink $\bot$, with capacities

$$c(a_2, \bot) \triangleq \mu_2(a_2)$$
$$c(\star_2, \bot) \triangleq \exp(-\varepsilon) \cdot \delta.$$

For the internal nodes, we connect $(a_1, a_2) \in \mathcal{A}_1 \times \mathcal{A}_2$ for all $(a_1, a_2) \in \mathcal{R}$ and $(a_1, \star_2)$, $(\star_1, a_2)$ for all a_1, a_2, all with capacity ∞.

Note that $(\{s\}, V \setminus \{s\})$ and $(V \setminus \{t\}, \{t\})$ are both cuts with capacity ω. We show that these are minimal cuts in the graph. Consider any other cut $C = (V_1, V_2)$ with edges $E(C)$ crossing the cut. If there is any internal edge $(a, b) \in E(C)$ with $a, b \neq \top, \bot$ then C has infinite capacity and is not a minimal cut. So, we may suppose $E(C)$ contains only edges of the form $(\top, a_1)$ and $(a_2, \bot)$.

Now if $E(C)$ does not contain $(\top, \star_1)$, then it must cut all edges leading into $\bot$; similarly, if $E(C)$ does not contain $(\star_2, \bot)$, then it must cut all edges leading from $\top$. Either way, its capacity is at least ω.

Finally, suppose $E(C)$ contains no internal edges and contains both $(\top, \star_1)$ and $(\star_2, \bot)$. Let $\mathcal{S}_2 \subseteq \mathcal{A}_2$ be the set of all nodes $a_2 \in \mathcal{A}_2$ with $(a_2, \bot) \in E(C)$, and let $\mathcal{S}_1 \subseteq \mathcal{A}_1$ be the set of all nodes $a_1 \in \mathcal{A}_1$ with $(\top, a_1) \in E(C)$. Since C separates $\top$ and $\bot$, we have

$$\mathcal{R}(\mathcal{A}_1 \setminus \mathcal{S}_1) \subseteq \mathcal{S}_2.$$

We can now lower-bound the capacity:

$$
\begin{aligned}
|C| &= c(\top, \mathcal{S}_1) + c(\mathcal{S}_2, \bot) + c(\top, \star_1) + c(\star_2, \bot) \\
&= \exp(-\varepsilon) \cdot \mu_1(\mathcal{S}_1) + c(\mathcal{S}_2, \bot) + \omega - \exp(-\varepsilon) \cdot \mu_1(\mathcal{A}_1) + \exp(-\varepsilon) \cdot \delta \\
&\geq \exp(-\varepsilon) \cdot \mu_1(\mathcal{S}_1) + c(\mathcal{R}(\mathcal{A}_1 \setminus \mathcal{S}_1), \bot) + \omega - \exp(-\varepsilon) \cdot \mu_1(\mathcal{A}_1) + \exp(-\varepsilon) \cdot \delta \\
&\geq \exp(-\varepsilon) \cdot \mu_1(\mathcal{S}_1) + \exp(-\varepsilon) \cdot \mu_1(\mathcal{A}_1 \setminus \mathcal{S}_1) - \exp(-\varepsilon) \cdot \delta + \omega - \exp(-\varepsilon) \cdot \mu_1(\mathcal{A}_1) + \exp(-\varepsilon) \cdot \delta \\
&= \omega
\end{aligned}
$$

The final inequality is by Sato's condition applied to the set $\mathcal{A}_1 \setminus \mathcal{S}_1$. So every cut in this graph has capacity at least ω, and there is a cut achieving capacity ω. By Theorem 5.1.3, there is a maximum flow f with weight ω. We define witnesses

$$
\begin{aligned}
\mu_L(a_1, a_2) &\triangleq \exp(\varepsilon) \cdot f(a_1, a_2) &&: \text{if } (a_1, a_2) \in \mathcal{R} \text{ or } a_2 = \star_2 \\
\mu_R(a_1, a_2) &\triangleq f(a_1, a_2) &&: \text{if } (a_1, a_2) \in \mathcal{R} \text{ or } a_1 = \star_1
\end{aligned}
$$

and zero otherwise. The support condition is clear. Since f has weight ω, it must saturate all edges exiting $\top$ and entering $\bot$ and so the marginal conditions are also clear.

The only thing to check is the distance condition $d_\varepsilon(\mu_L, \mu_R) \leq \delta$. It suffices to show this condition pointwise, by finding non-negative $\zeta(a_1, a_2)$ such that $\mu_L(a_1, a_2) \leq \exp(\varepsilon) \cdot \mu_R(a_1, a_2) + \zeta(a_1, a_2)$ and $\sum_{(a_1, a_2)} \zeta(a_1, a_2) \leq \delta$. For all $a_1 \in \mathcal{A}_1^\star$ and all $a_2 \neq \star_2$, we take $\zeta(a_1, a_2) = 0$. When $a_2 = \star_2$ we know

$$\mu_L(a_1, \star_2) = \exp(\varepsilon) \cdot f(a_1, \star_2) \qquad \text{and} \qquad \mu_R(a_1, \star_2) = 0,$$

so we may take $\zeta(a_1, \star_2) = \exp(\varepsilon) \cdot f(a_1, \star_2)$. Conservation of flow yields

$$\sum_{(a_1, a_2) \in \mathcal{A}_1^\star \times \mathcal{A}_2^\star} \zeta(a_1, a_2) = \sum_{a_1 \in \mathcal{A}_1} \exp(\varepsilon) \cdot f(a_1, \star_2) = \exp(\varepsilon) \cdot f(\star_2, \bot) = \delta,$$

establishing the desired distance condition $d_\varepsilon(\mu_L, \mu_R) \leq \delta$. $\qquad\square$

The countable case

There are several possible approaches to generalize Theorem 5.1.4 to countable distributions. Perhaps the most straightforward is to apply a version of the max flow-min cut theorem for countable graphs (Aharoni, Berger, Georgakopoulos, Perlstein, and Sprüssel, 2011). Instead, we will give a more elementary proof. Besides being self-contained, our proof also establishes limit and compactness properties of approximate couplings and their witnesses, which may be of independent interest.

   We first show that given a convergent sequence of pairs of distributions with an approximate lifting for each pair, there is a sub-sequence of witnesses converging to witnesses of an approximate lifting for the limits. We then generalize Theorem 5.1.4 to countable domains by viewing a distribution over a countable set as the pointwise limit of distributions with finite support, using the finite case to build approximate liftings (and witnesses) for each pair of finite restrictions

   We will need a generalized version of the dominated convergence theorem.

Theorem 5.1.5 (see, e.g., Royden and Fitzpatrick (2010, Chapter 4, Theorem 19)). *Let Ω be a measurable space with measure μ. Let $\{f_n\}$ and $\{g_n\}$ be two sequences of measurable functions $\Omega \to \mathbb{R}$ such that there exist functions $f, g : \Omega \to \mathbb{R}$ with*

   1. $\lim_{n \to \infty} f_n = f$ pointwise;

   2. $|f_n| \leq g_n$; and

   3. $\lim_{n \to \infty} \int g_n \, d\mu = \int g \, d\mu < \infty$.

Then we have

$$\lim_{n \to \infty} \int f_n \, d\mu = \int f \, d\mu.$$

   Since we work with countable spaces, we take μ to be the discrete measure. In this case, the integrals are simply plain sums. We will also need a lemma about witnesses to approximate liftings—roughly speaking, we may assume the witnesses are within a purely multiplicative factor of each other except on pairs with $\star$.

Lemma 5.1.6. *Suppose μ_1, μ_2 are sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$ such that*

$$\mu_1 \, \mathcal{R}^{\sharp(\varepsilon, \delta)} \, \mu_2.$$

Then there exists (η_L, η_R) witnessing the approximate lifting with

$$\eta_R(a_1, a_2) \leq \eta_L(a_1, a_2) \leq \exp(\varepsilon) \cdot \eta_R(a_1, a_2)$$

for all $a_1, a_2 \neq \star$.

Proof. Let μ_L, μ_R be witnesses. Define witnesses

$$\eta_L(a_1, a_2) \triangleq \begin{cases} \min(\mu_L(a_1, a_2), \exp(\varepsilon) \cdot \mu_R(a_1, a_2)) & : a_1 \neq \star, a_2 \neq \star \\ \mu_1(a_1) - \sum_{a_2' \in \mathcal{A}_2} \eta_L(a_1, a_2') & : a_1 \neq \star, a_2 = \star \\ 0 & : \text{otherwise;} \end{cases}$$

$$\eta_R(a_1, a_2) \triangleq \begin{cases} \min(\mu_L(a_1, a_2), \mu_R(a_1, a_2)) & : a_1 \neq \star, a_2 \neq \star \\ \mu_2(a_2) - \sum_{a_1' \in \mathcal{A}_1} \eta_R(a_1', a_2) & : a_1 = \star, a_2 \neq \star \\ 0 & : \text{otherwise.} \end{cases}$$

The marginal and support conditions follow from the respective conditions for (μ_L, μ_R). Note that η_L and η_R are non-negative by the marginal conditions for μ_L and μ_R. Furthermore for all $(a_1, a_2) \in \mathcal{A}_1 \times \mathcal{A}_2$, we have

$$\eta_R(a_1, a_2) \leq \eta_L(a_1, a_2) \leq \exp(\varepsilon) \cdot \eta_R(a_1, a_2).$$

It only remains to check the distance condition. Define non-negative constants

$$\zeta(a_1, a_2) \triangleq \max(\mu_L(a_1, a_2) - \exp(\varepsilon) \cdot \mu_R(a_1, a_2), 0).$$

Since $d_\varepsilon(\mu_L, \mu_R) \leq \delta$, we know $\mu_L(a_1, a_2) \leq \exp(\varepsilon) \cdot \mu_R(a_1, a_2) + \zeta(a_1, a_2)$ with equality when $\zeta(a_1, a_2) > 0$, and $\sum_{a_1, a_2 \in \mathcal{A}_1^\star \times \mathcal{A}_2^\star} \zeta(a_1, a_2) \leq \delta$. Thus, $\eta_L(a_1, a_2) = \mu_L(a_1, a_2) - \zeta(a_1, a_2)$ for every $a_1, a_2 \neq \star$. Also, we know $\eta_L(a_1, a_2) \leq \exp(\varepsilon) \cdot \eta_R(a_1, a_2)$. Thus for any subset $\mathcal{S} \subseteq \mathcal{A}_1^\star \times \mathcal{A}_2^\star$, we have

$$\begin{aligned} \eta_L(\mathcal{S}) &\leq \exp(\varepsilon) \cdot \eta_R(\mathcal{S} \cap (\mathcal{A}_1 \times \mathcal{A}_2)) + \eta_L(\mathcal{S} \cap (\mathcal{A}_1 \times \{\star\})) \\ &\leq \exp(\varepsilon) \cdot \eta_R(\mathcal{S} \cap (\mathcal{A}_1 \times \mathcal{A}_2)) + \eta_L(\mathcal{A}_1 \times \{\star\}) \\ &= \exp(\varepsilon) \cdot \eta_R(\mathcal{S} \cap (\mathcal{A}_1 \times \mathcal{A}_2)) + \sum_{a_1 \in \mathcal{A}_1} \left(\mu_1(a_1) - \sum_{a_2 \in \mathcal{A}_2} \mu_L(a_1, a_2) - \zeta(a_1, a_2) \right) \\ &= \exp(\varepsilon) \cdot \eta_R(\mathcal{S} \cap (\mathcal{A}_1 \times \mathcal{A}_2)) + \sum_{a_1 \in \mathcal{A}_1} \mu_L(a_1, \star) + \sum_{(a_1, a_2) \in \mathcal{A}_1 \times \mathcal{A}_2} \zeta(a_1, a_2) \\ &= \exp(\varepsilon) \cdot \eta_R(\mathcal{S} \cap (\mathcal{A}_1 \times \mathcal{A}_2)) + \sum_{(a_1, a_2) \in \mathcal{A}_1 \times \mathcal{A}_2^\star} \zeta(a_1, a_2) \\ &\leq \exp(\varepsilon) \cdot \eta_R(\mathcal{S}) + \delta. \end{aligned}$$ $\qquad \square$

We are now ready to prove that a converging sequence of pairs of distributions related by approximate liftings implies an approximate lifting for the limit distributions.

Lemma 5.1.7. *Let $\mathcal{R}$ be a binary relation between countable sets $\mathcal{A}_1, \mathcal{A}_2$. Consider a sequence $\{(\mu_1^{(n)}, \mu_2^{(n)})\}_{n \in \mathbb{N}}$ with $\mu_1^{(n)} \in \mathbf{SDistr}(\mathcal{A}_1)$ and $\mu_2^{(n)} \in \mathbf{SDistr}(\mathcal{A}_2)$ such that there exists an approximate lifting for each n:*

$$\mu_1^{(n)} \, \mathcal{R}^{\sharp(\varepsilon_n, \delta_n)} \, \mu_2^{(n)}.$$

Suppose $\lim_{n \to \infty} (\varepsilon_n, \delta_n) = (\varepsilon, \delta)$ and $\{\mu_1^{(n)}\}_n, \{\mu_2^{(n)}\}_n$ converge to μ_1, μ_2 under the L^1 norm:

$$\lim_{n \to \infty} \sum_{a_i \in \mathcal{A}_i} \left| \mu_i^{(n)}(a_i) - \mu_i(a_i) \right| = 0$$

for $i = 1, 2$. Then there exists an approximate lifting of the limit sub-distributions:

$$\mu_1 \, \mathcal{R}^{\sharp(\varepsilon, \delta)} \, \mu_2.$$

Proof. Let $(\eta_L^{(n)}, \eta_R^{(n)})$ witness the approximate lifting of $\mu_1^{(n)}$ and $\mu_2^{(n)}$, satisfying Lemma 5.1.6. Each witness can be viewed as a map $\eta_L^{(n)}, \eta_R^{(n)} : \mathcal{A}_1^\star \times \mathcal{A}_2^\star \to [0, 1]$. Since $\mathcal{A}_1$ and $\mathcal{A}_2$ are countable and $[0, 1]$ is compact, $\mathcal{A}_1^\star \times \mathcal{A}_2^\star \to [0, 1]$ is the countable product of compact sets and is itself (sequentially) compact. Hence, there exists

a sub-sequence of indices $\{\omega_n\}_n$ such that $\eta_L^{(\omega_n)}, \eta_R^{(\omega_n)}$ both converge pointwise to sub-distributions (η_L, η_R). (See any real analysis textbook, e.g., Royden and Fitzpatrick (2010) for a discussion about sequential compactness.)

We claim these limit sub-distributions are the desired witnesses. It is clear that $\text{supp}(\eta_L)$ and $\text{supp}(\eta_R)$ are contained in $\mathcal{R}$. The marginal conditions are a bit trickier. Let $a_1 \in \mathcal{A}_1$ (the marginal for $a_1 = \star$ is clear), and let ε_{max} be an upper bound of the sequence $\{\varepsilon_n\}_n$; since the sequence converges to ε, we may assume ε_{max} is finite. By Lemma 5.1.6 and the marginal condition on $\mu_2^{(\omega_n)}$, the sequence $\{\eta_L^{(\omega_n)}(a_1, -)\}_{n \in \mathbb{N}}$ is bounded by $\beta_L^{(\omega_n)} : \mathcal{A}_2^\star \to \mathbb{R}$, where

$$\beta_L^{(\omega_n)}(a_2) \triangleq \begin{cases} e^{\varepsilon_{max}} \mu_2^{(\omega_n)}(a_2) & : \text{if } a_2 \neq \star \\ 1 & : \text{if } a_2 = \star. \end{cases}$$

The sequence $\{\beta_L^{(\omega_n)}\}_n$ converges under the L^1 norm to $\beta_L : \mathcal{A}_2^\star \to \mathbb{R}$, where

$$\beta_L(a_2) \triangleq \begin{cases} e^{\varepsilon_{max}} \mu_2(a_2) & : \text{if } a_2 \neq \star \\ 1 & : \text{if } a_2 = \star. \end{cases}$$

Evidently $\sum_{a_2 \in \mathcal{A}_2^\star} \beta_L(a_2)$ exists and is at most $1 + e^{\varepsilon_{max}}$. Now for the first marginal,

$$\begin{aligned} \pi_1(\eta_L)(a_1) = \sum_{a_2 \in \mathcal{A}_2^\star} \eta_L(a_1, a_2) &= \sum_{a_2 \in \mathcal{A}_2^\star} \lim_{n \to \infty} \eta_L^{(\omega_n)}(a_1, a_2) \\ &= \lim_{n \to \infty} \sum_{a_2 \in \mathcal{A}_2^\star} \eta_L^{(\omega_n)}(a_1, a_2) = \lim_{n \to \infty} \pi_1(\eta_L^{(\omega_n)})(a_1) \\ &= \lim_{n \to \infty} \mu_1^{(\omega_n)}(a_1) = \mu_1(a_1). \end{aligned}$$

We can interchange the sum and the limit by the dominated convergence theorem with bounding functions $\beta_L^{(\omega_n)}$ (Theorem 5.1.5).

For the second marginal, let $a_2 \in \mathcal{A}_2$ (the marginal for $a_2 = \star$ is clear). By Lemma 5.1.6 and the marginal condition on $\mu_1^{(\omega_n)}$, the sequence $\{\eta_R^{(\omega_n)}(-, a_2)\}_{n \in \mathbb{N}}$ is bounded by $\beta_R^{(\omega_n)} : \mathcal{A}_1^\star \to \mathbb{R}$, where

$$\beta_R^{(\omega_n)}(a_1) \triangleq \begin{cases} \mu_1^{(\omega_n)}(a_1) & : \text{if } a_1 \neq \star \\ 1 & : \text{if } a_1 = \star. \end{cases}$$

The sequence $\{\beta_R^{(\omega_n)}\}_n$ converges under the L^1 norm to $\beta_R : \mathcal{A}_1^\star \to \mathbb{R}$, where

$$\beta_R(a_1) \triangleq \begin{cases} \mu_1(a_1) & : \text{if } a_1 \neq \star \\ 1 & : \text{if } a_1 = \star. \end{cases}$$

Evidently $\sum_{a_1 \in \mathcal{A}_1^\star} \beta_R(a_1)$ exists and is at most 2. For the second marginal,

$$\begin{aligned} \pi_2(\eta_R)(a_2) = \sum_{a_1 \in \mathcal{A}_1^\star} \eta_R(a_1, a_2) &= \sum_{a_1 \in \mathcal{A}_1^\star} \lim_{n \to \infty} \eta_R^{(\omega_n)}(a_1, a_2) \\ &= \lim_{n \to \infty} \sum_{a_1 \in \mathcal{A}_1^\star} \eta_R^{(\omega_n)}(a_1, a_2) = \lim_{n \to \infty} \pi_2(\eta_R^{(\omega_n)})(a_2) \\ &= \lim_{n \to \infty} \mu_2^{(\omega_n)}(a_2) = \mu_2(a_2). \end{aligned}$$

As before, to interchange the sum and the limit we apply the dominated convergence theorem with bounding functions $\beta_R^{(\omega_n)}$ (Theorem 5.1.5).

The distance condition now follows by taking limits. For any subset $\mathcal{S} \subseteq \mathcal{A}_1^\star \times \mathcal{A}_2^\star$, we have

$$
\begin{aligned}
\eta_L(\mathcal{S}) - \exp(\varepsilon) \cdot \eta_R(\mathcal{S}) &= \lim_{n \to \infty} \eta_L^{(\omega_n)}(\mathcal{S}) - \lim_{n \to \infty} \exp(\varepsilon_{\omega_n}) \cdot \lim_{n \to \infty} \eta_R^{(\omega_n)}(\mathcal{S}) \\
&= \lim_{n \to \infty} \left(\eta_L^{(\omega_n)}(\mathcal{S}) - \exp(\varepsilon_{\omega_n}) \cdot \eta_R^{(\omega_n)}(\mathcal{S}) \right) \\
&\leq \lim_{n \to \infty} \delta_{\omega_n} \\
&= \delta.
\end{aligned}
$$
$\square$

Finally, we obtain the countable version of Theorem 5.1.4.

Theorem 5.1.8. *Let μ_1 and μ_2 be sub-distributions over countable sets $\mathcal{A}_1$ and $\mathcal{A}_2$, and let $\mathcal{R} \subseteq \mathcal{A}_1 \times \mathcal{A}_2$ be a binary relation such that $\mu_1(\mathcal{S}_1) \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{R}(\mathcal{S}_1)) + \delta$ for every $\mathcal{S}_1 \subseteq \mathcal{A}_1$. Then there exists an approximate lifting*

$$
\mu_1 \, \mathcal{R}^{\sharp(\varepsilon, \delta)} \, \mu_2.
$$

Proof. Since $\mathcal{A}_1$ and $\mathcal{A}_2$ are countable, there are finite subsets $\mathcal{I}_1^{(n)} \subseteq \mathcal{A}_1, \mathcal{I}_2^{(n)} \subseteq \mathcal{A}_2$ such that $\{\mathcal{I}_1^{(n)}\}_n$ and $\{\mathcal{I}_2^{(n)}\}_n$ are increasing with $\cup_n \mathcal{I}_1^{(n)} = \mathcal{A}_1$ and $\cup_n \mathcal{I}_2^{(n)} = \mathcal{A}_2$. Consider the sequences of restricted sub-distributions

$$
\mu_1^{(n)}(a_1) \triangleq \begin{cases} \mu_1(a_1) & : a_1 \in \mathcal{I}_1^{(n)} \\ 0 & : \text{otherwise} \end{cases} \qquad \mu_2^{(n)}(a_2) \triangleq \begin{cases} \mu_2(a_2) & : a_2 \in \mathcal{I}_2^{(n)} \\ 0 & : \text{otherwise.} \end{cases}
$$

For any subset $\mathcal{S}_1 \subseteq \mathcal{A}_1$, by assumption

$$
\mu_1(\mathcal{S}_1) \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{R}(\mathcal{S}_1)) + \delta.
$$

On the restricted sub-distributions, we have

$$
\begin{aligned}
\mu_1^{(n)}(\mathcal{S}_1) \leq \mu_1(\mathcal{S}_1) &\leq \exp(\varepsilon) \cdot \mu_2(\mathcal{R}(\mathcal{S}_1) \cap \mathcal{I}_2^{(n)}) + \exp(\varepsilon) \cdot \mu_2(\mathcal{A}_2 \setminus \mathcal{I}_2^{(n)}) + \delta \\
&\triangleq \exp(\varepsilon) \cdot \mu_2^{(n)}(\mathcal{R}(\mathcal{S}_1)) + \delta_n.
\end{aligned}
$$

Evidently $\lim_{n \to \infty} \delta_n = \delta$. Since $\mu_1^{(n)}$ and $\mu_2^{(n)}$ have finite support contained in $\mathcal{I}_1^{(n)}$ and $\mathcal{I}_2^{(n)}$, Theorem 5.1.4 gives an approximate lifting for each finite restriction:

$$
\mu_1^{(n)} \, \mathcal{R}^{\sharp(\varepsilon, \delta_n)} \, \mu_2^{(n)}.
$$

Since $\mu_1^{(n)}$ and $\mu_2^{(n)}$ converge in L^1 to μ_1 and μ_2, we can conclude by Lemma 5.1.7. $\square$

Alternative proofs of coupling constructions

The equivalence from Theorems 5.1.2 and 5.1.8 gives a convenient way to construct approximate couplings. For instance, we can easily prove a transitivity principle.

Lemma 5.1.9. *Let μ_1, μ_2, μ_3 be sub-distributions over $\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$ respectively, and let $\mathcal{R} \subseteq \mathcal{A}_1 \times \mathcal{A}_2$ and $\mathcal{S} \subseteq \mathcal{A}_2 \times \mathcal{A}_3$ be binary relations. If we have*

$$
\mu_1 \, \mathcal{R}^{\sharp(\varepsilon, \delta)} \, \mu_2 \qquad \text{and} \qquad \mu_2 \, \mathcal{S}^{\sharp(\varepsilon', \delta')} \, \mu_3,
$$

then we also have

$$
\mu_1 \, (\mathcal{S} \circ \mathcal{R})^{\sharp(\varepsilon + \varepsilon', \exp(\varepsilon')\delta + \delta')} \, \mu_3.
$$

Proof. Let $\mathcal{T}_1 \subseteq \mathcal{A}_1$ be any subset. By Theorem 5.1.2 we have

$$\mu_1(\mathcal{T}_1) \leq \exp(\varepsilon)\mu_2(\mathcal{R}(\mathcal{T}_1)) + \delta$$
$$\mu_2(\mathcal{R}(\mathcal{T}_1)) \leq \exp(\varepsilon')\mu_3((\mathcal{S} \circ \mathcal{R})(\mathcal{T}_1)) + \delta'.$$

Chaining the inequalities and applying Theorem 5.1.8 yields the desired approximate lifting. □

We can also give alternative proofs for the couplings from Chapter 4.

Theorem 4.2.7. *Let μ_1 and μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$. If we have functions $f_i : \mathcal{A}_i \to \mathcal{B}_i$ for $i \in \{1, 2\}$, and a relation $\mathcal{R} \subseteq \mathcal{B}_1 \times \mathcal{B}_2$, then*

$$\mu_1 \; \{(a_1, a_2) \in \mathcal{A}_1 \times \mathcal{A}_2 \mid f_1(a_1) \; \mathcal{R} \; f_2(a_2)\}^{\sharp(\varepsilon, \delta)} \; \mu_2$$

if and only if

$$f_1^{\sharp}(\mu_1) \; \{(b_1, b_2) \in \mathcal{B}_1 \times \mathcal{B}_2 \mid b_1 \; \mathcal{R} \; b_2\}^{\sharp(\varepsilon, \delta)} \; f_2^{\sharp}(\mu_2).$$

(Recall $f : \mathcal{A} \to \mathcal{B}$ can be lifted to a map $f^{\sharp} : \mathbf{SDistr}(\mathcal{A}) \to \mathbf{SDistr}(\mathcal{B})$ on sub-distributions.)

Proof (alternative). For the forward direction, let $\mathcal{T}_1 \subseteq \mathcal{B}_1$ be any subset. Then

$$
\begin{aligned}
f_1^{\sharp}(\mu_1)(\mathcal{T}_1) &= \mu_1(f_1^{-1}(\mathcal{T}_1)) \\
&\leq \exp(\varepsilon) \cdot \mu_2(f_2^{-1}(\mathcal{R}(\mathcal{T}_1))) + \delta \qquad \text{(Theorem 5.1.2)} \\
&= \exp(\varepsilon) \cdot f_2^{\sharp}(\mu_2)(\mathcal{R}(\mathcal{T}_1)) + \delta,
\end{aligned}
$$

so we conclude by Theorem 5.1.8. For the reverse direction, let $\mathcal{S}_1 \subseteq \mathcal{A}_1$ be any subset. Then

$$
\begin{aligned}
\mu_1(\mathcal{S}_1) &\leq \mu_1(f_1^{-1}(f_1(\mathcal{S}_1))) \\
&= f_1^{\sharp}(\mu_1)(f_1(\mathcal{S}_1)) \\
&\leq \exp(\varepsilon) \cdot f_2^{\sharp}(\mu_2)(\mathcal{R}(f_1(\mathcal{S}_1))) + \delta \qquad \text{(Theorem 5.1.2)} \\
&= \exp(\varepsilon) \cdot \mu_2(f_2^{-1}(\mathcal{R}(f_1(\mathcal{S}_1)))) + \delta.
\end{aligned}
$$

Since $f_1(x_1) \; \mathcal{R} \; f_2(x_2)$ precisely when $x_1 \; (f_2^{-1} \circ \mathcal{R} \circ f_1) \; x_2$, we conclude by Theorem 5.1.8. □

Proposition 4.5.1. *Let $v_1, v_2 \in \mathbb{Z}$. Then:*

$$\mathbf{Lap}_{\varepsilon}(v_1) \; \{(x_1, x_2) \mid x_1 - v_1 = x_2 - v_2\}^{\sharp(0, 0)} \; \mathbf{Lap}_{\varepsilon}(v_2).$$

Proof (alternative). Let $\mathcal{S} \subseteq \mathbb{Z}$ be any subset and let $\mathcal{S}'$ be the set $\{s - v_1 + v_2 \mid s \in \mathcal{S}\}$. Noting $\mathbf{Lap}_{\varepsilon}(v_1)(s) = \mathbf{Lap}_{\varepsilon}(v_2)(s - v_1 + v_2)$ for every s and summing over all $s \in \mathcal{S}$, we have

$$\mathbf{Lap}_{\varepsilon}(v_1)(\mathcal{S}) = \mathbf{Lap}_{\varepsilon}(v_2)(\mathcal{S}').$$

Theorem 5.1.8 gives the desired approximate coupling. □

Proposition 4.5.3. *Let $k, k', v_1, v_2 \in \mathbb{Z}$, and suppose $|k + v_1 - v_2| \leq k'$. Then:*

$$\mathbf{Lap}_{\varepsilon}(v_1) \; \{(x_1, x_2) \mid x_1 + k = x_2\}^{\sharp(k'\varepsilon, 0)} \; \mathbf{Lap}_{\varepsilon}(v_2).$$

Proof (alternative). Let $\mathcal{S} \subseteq \mathbb{Z}$ be any subset and let $\mathcal{S}'$ be the set $\{s + k \mid s \in \mathcal{S}\}$. Noting

$$\mathbf{Lap}_\varepsilon(v_1)(s) = \mathbf{Lap}_\varepsilon(v_2)(s - v_1 + v_2)$$
$$\leq \exp(|k - v_2 + v_1|\varepsilon) \cdot \mathbf{Lap}_\varepsilon(v_2)(s + k)$$
$$\leq \exp(k'\varepsilon) \cdot \mathbf{Lap}_\varepsilon(v_2)(s + k)$$

for every s and summing over all $s \in \mathcal{S}$, we have

$$\mathbf{Lap}_\varepsilon(v_1)(\mathcal{S}) = \exp(k'\varepsilon) \cdot \mathbf{Lap}_\varepsilon(v_2)(\mathcal{S}').$$

Theorem 5.1.8 gives the desired approximate coupling. □

Proposition 4.6.1. *Let μ_1, μ_2 be sub-distributions over $\mathcal{R}$ and suppose for every $i \in \mathcal{R}$, we have*

$$\mu_1 \{(r_1, r_2) \mid r_1 = i \to r_2 = i\}^{\sharp(\varepsilon, \delta_i)} \mu_2$$

for non-negative ε and $\{\delta_i\}_{i \in \mathcal{R}}$. Then we have

$$\mu_1 \, (=)^{\sharp(\varepsilon, \delta)} \, \mu_2$$

where $\delta = \sum_{i \in \mathcal{R}} \delta_i$.

Proof (alternative). By Theorem 5.1.2 we have $\mu_1(i) \leq \exp(\varepsilon) \cdot \mu_2(i) + \delta_i$ for every $i \in \mathcal{R}$. Hence for any set $\mathcal{S} \subseteq \mathcal{R}$, summing over $i \in \mathcal{S}$ gives

$$\mu_1(\mathcal{S}) \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{S}) + \sum_{i \in \mathcal{S}} \delta_i \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{S}) + \delta.$$

Theorem 5.1.8 gives the desired approximate coupling. □

For the couplings we introduce in the rest of this chapter, we will give each construction in two ways: first as a consequence of Sato's definition, then in terms of two explicit witness distributions.

5.2 Accuracy-dependent approximate couplings

A common technique in proofs for cryptographic protocols is *up-to-bad* reasoning. Roughly, two versions of a protocol—say, one that operates on the true secret information and one that operates on random noise—are said to be *equivalent up-to-bad* if they have the same distribution over outputs assuming some probabilistic event, the so-called *bad event*, does not happen. If the bad event has small probability, up-to-bad equivalence implies that the output distributions of the two programs are close. This principle can be seen as a property about exact couplings, a consequence of the coupling method (Theorem 2.1.16).

Proposition 5.2.1. *Let μ_1, μ_2 be sub-distributions over $\mathcal{A}$ and let $\mathcal{P} \subseteq \mathcal{A}$ be a subset. If for $i \in \{1, 2\}$ we have an exact lifting*

$$\mu_1 \{(x_1, x_2) \mid x_i \in \mathcal{P} \to x_1 = x_2\}^\sharp \mu_2,$$

then $d_{tv}(\mu_1, \mu_2) \leq \mu_i(\mathcal{A} \setminus \mathcal{P})$.

Proof. Let μ be the witness. We have

$$d_{tv}(\mu_1, \mu_2) \leq \Pr_{(x_1, x_2) \sim \mu}[x_1 \neq x_2] = \Pr_{(x_1, x_2) \sim \mu}[x_1 \neq x_2 \wedge x_i \notin \mathcal{P}] \leq \mu_i(\mathcal{A} \setminus \mathcal{P}),$$

by Theorem 2.1.16, the support condition, and the first marginal condition. □

Up-to-bad approximate couplings

The δ parameter of an approximate coupling is closely related to TV-distance. For example, the distance bound $d_0(\mu_1, \mu_2) \leq \delta$ is equivalent to $d_{\mathrm{tv}}(\mu_1, \mu_2) \leq \delta$ for proper distributions. This observation suggests we can generalize Proposition 5.2.1 to approximate couplings. We introduce two constructions, which we call *up-to-bad approximate couplings*.

Proposition 5.2.2. *Let μ_1, μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$, and let $\mathcal{P}_1, \mathcal{P}_2$ be subsets of $\mathcal{A}_1$ and $\mathcal{A}_2$. Consider any binary relation $\mathcal{R} \subseteq \mathcal{A}_1 \times \mathcal{A}_2$.*

1. *If $\mu_1(\mathcal{A}_1 \setminus \mathcal{P}_1) \leq \delta'$, then*

$$\mu_1 \; \{(a_1, a_2) \mid a_1 \in \mathcal{P}_1 \to (a_1, a_2) \in \mathcal{R}\}^{\sharp(\varepsilon, \delta)} \; \mu_2 \qquad \text{implies} \qquad \mu_1 \; \mathcal{R}^{\sharp(\varepsilon, \delta + \delta')} \; \mu_2.$$

2. *If $\mu_2(\mathcal{A}_2 \setminus \mathcal{P}_2) \leq \delta'$, then*

$$\mu_1 \; \{(a_1, a_2) \mid a_2 \in \mathcal{P}_2 \to (a_1, a_2) \in \mathcal{R}\}^{\sharp(\varepsilon, \delta)} \; \mu_2 \qquad \text{implies} \qquad \mu_1 \; \mathcal{R}^{\sharp(\varepsilon, \delta + \exp(\varepsilon) \cdot \delta')} \; \mu_2.$$

The slight difference between the two versions is due to our asymmetric definition of approximate coupling; bad events in μ_1 are not treated the same as bad events in μ_2.

Proof. We first introduce some notation for binary relations and sets. First, we will interpret $\mathcal{P}_1$ and $\mathcal{P}_2$ as subsets of $\mathcal{A}_1 \times \mathcal{A}_2$ via $\mathcal{P}_1 \times \mathcal{A}_2$ and $\mathcal{A}_1 \times \mathcal{P}_2$. If $\mathcal{R}$ is a binary relation over $\mathcal{B}_1 \times \mathcal{B}_2$, we write $\neg \mathcal{R}$ for the binary relation $\mathcal{B}_1 \times \mathcal{B}_2 \setminus \mathcal{R}$. Finally, we write $\mathcal{A} \to \mathcal{B}$ for the binary relation $\neg \mathcal{B} \cup \mathcal{A}$.

To prove the first point, let $\mathcal{S}_1 \subseteq \mathcal{A}_1$ be any subset. By assumption and Theorem 5.1.2,

$$\mu_1(\mathcal{S}_1 \cap \mathcal{P}_1) \leq \exp(\varepsilon) \cdot \mu_2((\mathcal{P}_1 \to \mathcal{R})(\mathcal{S}_1 \cap \mathcal{P}_1)) + \delta = \exp(\varepsilon) \cdot \mu_2(\mathcal{R}(\mathcal{S}_1 \cap \mathcal{P}_1)) + \delta.$$

Since $\mu_1(\neg \mathcal{P}_1) \leq \delta'$, we also have

$$\mu_1(\mathcal{S}_1) \leq \mu_1(\mathcal{S}_1 \cap \mathcal{P}_1) + \delta' \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{R}(\mathcal{S}_1 \cap \mathcal{P}_1)) + \delta + \delta' \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{R}(\mathcal{S}_1)) + \delta + \delta'$$

and hence Theorem 5.1.8 gives the desired approximate coupling.

The second point is similar. Let $\mathcal{S}_1 \subseteq \mathcal{A}_1$ be any subset. By assumption and Theorem 5.1.2,

$$\begin{aligned}
\mu_1(\mathcal{S}_1) &\leq \exp(\varepsilon) \cdot \mu_2((\mathcal{P}_2 \to \mathcal{R})(\mathcal{S}_1)) + \delta \\
&\leq \exp(\varepsilon) \cdot \mu_2(\neg \mathcal{P}_2) + \exp(\varepsilon) \cdot \mu_2(\mathcal{R}(\mathcal{S}_1)) + \delta \\
&\leq \exp(\varepsilon) \cdot \mu_2(\mathcal{R}(\mathcal{S}_1)) + \delta + \exp(\varepsilon) \cdot \delta'
\end{aligned}$$

and hence Theorem 5.1.8 gives the desired approximate coupling.

To give witnesses for the first point, let μ_L, μ_R witness the approximate lifting of $\mathcal{P}_1 \to \mathcal{R}$. We define two witnesses $\eta_L, \eta_R \in \mathbf{SDistr}(\mathcal{A}_1^\star \times \mathcal{A}_2^\star)$ for the approximate lifting of $\mathcal{R}$:

$$\eta_L(a_1, a_2) \triangleq \begin{cases} \mu_L(a_1, a_2) & : (a_1, a_2) \in \mathcal{R} \\ \mu_L(a_1, \star) + \sum_{a_2 \in \mathcal{A}_2 : (a_1, a_2) \notin \mathcal{R}} \mu_L(a_1, a_2) & : a_2 = \star \\ 0 & : \text{otherwise.} \end{cases}$$

$$\eta_R(a_1, a_2) \triangleq \begin{cases} \mu_R(a_1, a_2) & : (a_1, a_2) \in \mathcal{R} \\ \mu_R(\star, a_2) + \sum_{a_1' \in \mathcal{A}_1 : (a_1', a_2) \notin \mathcal{R}} \mu_R(a_1', a_2) & : a_1 = \star \\ 0 & : \text{otherwise.} \end{cases}$$

By construction, $\mathrm{supp}(\eta_L) \cup \mathrm{supp}(\eta_R) \subseteq \mathcal{R}^\star$. We can check the first marginal condition:

$$
\begin{aligned}
\pi_1(\eta_L)(a_1) &= \sum_{a_2 \in \mathcal{A}_2^\star} \eta_L(a_1, a_2) \\
&= \eta_L(a_1, \star) + \sum_{a_2 \in \mathcal{A}_2 : (a_1, a_2) \in \mathcal{R}} \eta_L(a_1, a_2) \\
&= \mu_L(a_1, \star) + \sum_{a_2 \in \mathcal{A}_2 : (a_1, a_2) \notin \mathcal{R}} \mu_L(a_1, a_2) + \sum_{a_2 \in \mathcal{A}_2 : (a_1, a_2) \in \mathcal{R}} \mu_L(a_1, a_2) \\
&= \sum_{a_2 \in \mathcal{A}_2^\star} \mu_L(a_1, a_2) = \pi_1(\mu_L)(a_1).
\end{aligned}
$$

The second marginal is similar:

$$
\begin{aligned}
\pi_2(\eta_R)(a_2) &= \sum_{a_1 \in \mathcal{A}_1^\star} \eta_R(a_1, a_2) \\
&= \eta_R(\star, a_2) + \sum_{a_1 \in \mathcal{A}_1 : (a_1, a_2) \in \mathcal{R}} \eta_R(a_1, a_2) \\
&= \mu_R(\star, a_2) + \sum_{a_1 \in \mathcal{A}_1 : (a_1, a_2) \notin \mathcal{R}} \mu_R(a_1, a_2) + \sum_{a_1 \in \mathcal{A}_1 : (a_1, a_2) \in \mathcal{R}} \mu_R(a_1, a_2) \\
&= \sum_{a_1 \in \mathcal{A}_1^\star} \mu_R(a_1, a_2) = \pi_2(\mu_R)(a_2).
\end{aligned}
$$

It remains to check the distance condition. Compared to the old witnesses, the new witnesses have larger mass on subsets satisfying $\mathcal{R}^\star$: for all subsets $\mathcal{S} \subseteq \mathcal{R}^\star$, we have $\mu_L(\mathcal{S}) \leq \eta_L(\mathcal{S})$ and $\mu_R(\mathcal{S}) \leq \eta_R(\mathcal{S})$. For any set $\mathcal{S} \subseteq \mathcal{A}_1^\star \times \mathcal{A}_2^\star$, we can also bound $\eta_L(\mathcal{S})$ from above:

$$
\begin{aligned}
\eta_L(\mathcal{S}) &= \sum_{(a_1, a_2) \in \mathcal{S} \cap \mathcal{R}^\star} \eta_L(a_1, a_2) \\
&= \sum_{(a_1, a_2) \in \mathcal{S} \cap \mathcal{R}} \eta_L(a_1, a_2) + \sum_{(a_1, \star) \in \mathcal{S}} \eta_L(a_1, \star) \\
&= \sum_{(a_1, a_2) \in \mathcal{S} \cap \mathcal{R}} \mu_L(a_1, a_2) + \sum_{(a_1, \star) \in \mathcal{S}} \left(\mu_L(a_1, \star) + \sum_{a_2 \in \mathcal{A}_2 : (a_1, a_2) \notin \mathcal{R}} \mu_L(a_1, a_2) \right) \\
&\leq \sum_{(a_1, a_2) \in \mathcal{S}} \mu_L(a_1, a_2) + \sum_{a_1 \in \mathcal{A}_1 \setminus \mathcal{P}_1} \sum_{a_2 \in \mathcal{A}_2^\star} \mu_L(a_1, a_2) \\
&= \sum_{(a_1, a_2) \in \mathcal{S}} \mu_L(a_1, a_2) + \mu_1(\neg \mathcal{P}_1) \\
&\leq \mu_L(\mathcal{S}) + \delta'.
\end{aligned}
$$

The first inequality uses the support of μ_L; the final inequality is by assumption. Finally, we chain these bounds:

$$
\begin{aligned}
\eta_L(\mathcal{S}) &= \eta_L(\mathcal{S} \cap \mathcal{R}^\star) \\
&\leq \mu_L(\mathcal{S} \cap \mathcal{R}^\star) + \delta' \\
&\leq \exp(\varepsilon) \cdot \mu_R(\mathcal{S} \cap \mathcal{R}^\star) + \delta + \delta' \\
&\leq \exp(\varepsilon) \cdot \eta_R(\mathcal{S} \cap \mathcal{R}^\star) + \delta + \delta' \\
&= \exp(\varepsilon) \cdot \eta_R(\mathcal{S}) + \delta + \delta'.
\end{aligned}
$$

$$\text{UTB-L} \ \frac{\vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Theta\langle 1 \rangle \to \Psi \qquad \forall m, \models \Pr_{[\![c_1]\!]m} [\neg\Theta] \leq \delta'}{\vdash c_1 \sim_{(\varepsilon,\delta+\delta')} c_2 : \Phi \Longrightarrow \Psi}$$

$$\text{UTB-R} \ \frac{\vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Theta\langle 2 \rangle \to \Psi \qquad \forall m, \models \Pr_{[\![c_2]\!]m} [\neg\Theta] \leq \delta'}{\vdash c_1 \sim_{(\varepsilon,\delta+\exp(\varepsilon)\cdot\delta')} c_2 : \Phi \Longrightarrow \Psi}$$

Figure 5.1: Up-to-bad rules for APRHL

This implies $d_\varepsilon(\eta_L, \eta_R) \leq \delta + \delta'$, so η_L and η_R witness the approximate lifting.

To give witnesses for the second point, let η_L, η_R be defined as above and consider any subset $\mathcal{S} \subseteq \mathcal{A}_1^\star \times \mathcal{A}_2^\star$. The marginal and support conditions follow as before. To check the distance condition, we first bound η_L in terms of μ_L:

$$\begin{aligned}
\eta_L(\mathcal{S}) &= \sum_{(a_1,a_2)\in\mathcal{S}} \eta_L(a_1, a_2) \\
&\leq \sum_{(a_1,a_2)\in\mathcal{S}\cap\mathcal{R}^\star} \mu_L(a_1, a_2) + \sum_{(a_1,a_2)\notin\mathcal{R}} \mu_L(a_1, a_2) \\
&= \mu_L(\mathcal{S}\cap\mathcal{R}^\star) + \mu_L(\neg\mathcal{R}) \\
&= \mu_L((\mathcal{S}\cap\mathcal{R}^\star)\cup\neg\mathcal{R})
\end{aligned}$$

The last equality is because the two events are disjoint. We then complete the calculation as before:

$$\begin{aligned}
\eta_L(\mathcal{S}) &\leq \mu_L((\mathcal{S}\cap\mathcal{R}^\star)\cup\neg\mathcal{R}) \\
&\leq \exp(\varepsilon)\cdot\mu_R((\mathcal{S}\cap\mathcal{R}^\star)\cup\neg\mathcal{R}) + \delta \\
&\leq \exp(\varepsilon)(\mu_R(\mathcal{S}\cap\mathcal{R}^\star) + \mu_R(\neg\mathcal{P}_2)) + \delta \\
&= \exp(\varepsilon)(\mu_R(\mathcal{S}\cap\mathcal{R}^\star) + \mu_R(\neg\mathcal{P}_2)) + \delta \\
&\leq \exp(\varepsilon)(\mu_R(\mathcal{S}\cap\mathcal{R}^\star) + \delta') + \delta \\
&\leq \exp(\varepsilon)(\eta_R(\mathcal{S}\cap\mathcal{R}^\star) + \delta') + \delta \\
&= \exp(\varepsilon)\cdot\eta_R(\mathcal{S}) + \delta + \exp(\varepsilon)\cdot\delta'.
\end{aligned}$$

Thus $d_\varepsilon(\eta_L, \eta_R) \leq \delta + \exp(\varepsilon)\cdot\delta'$, so (η_L, η_R) witness the desired approximate coupling. $\qquad\square$

We realize these couplings in APRHL with the up-to-bad rules in Fig. 5.1. In both rules, Θ is a predicate on **State**; $\Theta\langle 1 \rangle$ and $\Theta\langle 2 \rangle$ are the associated predicates on the product memories **State**$_\times$; syntactically, where all variables in Θ are tagged with $\langle 1 \rangle$ or $\langle 2 \rangle$ respectively.

Theorem 5.2.3. *The rules* [UTB-L] *and* [UTB-R] *are sound.*

Proof. By validity of the premises and Proposition 5.2.2. $\qquad\square$

Figure 5.2 presents two useful variants of the up-to-bad rules that are restricted versions of the rule of conjunction from Hoare logic. As we discussed before, the general conjunction rule is not sound in PRHL, nor in APRHL. However if one of the conjuncts mentions only one side, we can recover a version of the conjunction rule.

Corollary 5.2.4. *The rules* [AND-L] *and* [AND-R] *are sound.*

$$\text{And-L} \; \frac{\vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi \qquad \forall m, \models \Pr_{[\![c_1]\!]m} [\neg\Theta] \le \delta'}{\vdash c_1 \sim_{(\varepsilon,\delta+\delta')} c_2 : \Phi \Longrightarrow \Theta\langle 1 \rangle \wedge \Psi}$$

$$\text{And-R} \; \frac{\vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi \qquad \forall m, \models \Pr_{[\![c_2]\!]m} [\neg\Theta] \le \delta'}{\vdash c_1 \sim_{(\varepsilon,\delta+\exp(\varepsilon)\cdot\delta')} c_2 : \Phi \Longrightarrow \Theta\langle 2 \rangle \wedge \Psi}$$

Figure 5.2: One-sided conjunction rules for APRHL

$$\text{LapAcc-L} \; \frac{\vdash x_1 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_1) \sim_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi \qquad x_1 \notin \mathrm{FV}(e_1)}{\vdash x_1 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_1) \sim_{(\varepsilon,\delta+\beta)} c_2 : \Phi \Longrightarrow |x_1\langle 1 \rangle - e_1\langle 1 \rangle| \le \frac{1}{\varepsilon} \ln \frac{1}{\beta} \wedge \Psi}$$

$$\text{LapAcc-R} \; \frac{\vdash c_1 \sim_{(\varepsilon,\delta)} x_2 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_2) : \Phi \Longrightarrow \Psi \qquad x_2 \notin \mathrm{FV}(e_2)}{\vdash c_1 \sim_{(\varepsilon,\delta+\exp(\varepsilon)\cdot\beta)} x_2 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_2) : \Phi \Longrightarrow |x_2\langle 2 \rangle - e_2\langle 2 \rangle| \le \frac{1}{\varepsilon} \ln \frac{1}{\beta} \wedge \Psi}$$

Figure 5.3: Laplace accuracy bounds in APRHL

Proof. From the premise of [AND-L], the rule of consequence gives

$$\vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Theta\langle 1 \rangle \rightarrow \Theta\langle 1 \rangle \wedge \Psi$$

and hence we can conclude by applying [UTB-L]:

$$\vdash c_1 \sim_{(\varepsilon,\delta+\delta')} c_2 : \Phi \Longrightarrow \Theta\langle 1 \rangle \wedge \Psi.$$

Similarly, we can derive [AND-R] from [UTB-R]. □

When $\delta' = 0$, the rules [AND-L] and [AND-R] can add one-sided support assertions to the post-condition of any APRHL rule. This can be useful to work around the narrow post-conditions in certain APRHL rules (e.g., [PW-EQ]). We can also use these rules to introduce accuracy bounds. As an example, we give a basic tail bound for the discrete Laplace distribution.

Proposition 5.2.5. *Let $\varepsilon, \beta > 0$ and let $t \in \mathbb{Z}$. Then we can bound the probability of samples from the Laplace distribution being far from the mean:*

$$\Pr_{x \sim \mathbf{Lap}_\varepsilon(t)} \left[|x - t| > \frac{1}{\varepsilon} \ln \frac{1}{\beta} \right] \le \beta.$$

This bound gives the two rules in Fig. 5.3.

Corollary 5.2.6. *The rules [LAPACC-L] and [LAPACC-R] are sound.*

Proof. By the rules [AND-L], [AND-R], and Proposition 5.2.5. □

5.3   Optimal subset coupling

By Proposition 4.2.4, approximate lifted implication

$$\mu_1 \{(a_1, a_2) \mid a_1 \in \mathcal{S}_1 \rightarrow a_2 \in \mathcal{S}_2\}^{\sharp(\varepsilon, \delta)} \mu_2$$

ensures $\mu_1(\mathcal{S}_1) \leq \exp(\varepsilon) \cdot \mu_2(\mathcal{S}_2) + \delta$. In this section, we explore a partial converse.

Theorem 5.3.1 (Optimal subset coupling). *Let $\alpha \geq 1$ and $\delta \geq 0$. Let μ_1 and μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$ with equal weight, and consider subsets $\mathcal{S}_1 \subseteq \mathcal{A}_1, \mathcal{S}_2 \subseteq \mathcal{A}_2$. Then $\mu_1(\mathcal{S}_1) \leq \alpha\mu_2(\mathcal{S}_2) + \delta$ and $\mu_1(\mathcal{A}_1 \setminus \mathcal{S}_1) \leq \alpha\mu_2(\mathcal{A}_2 \setminus \mathcal{S}_2) + \delta$ if and only if*

$$\mu_1 \{(a_1, a_2) \mid a_1 \in \mathcal{S}_1 \leftrightarrow a_2 \in \mathcal{S}_2\}^{\sharp(\ln \alpha, \delta)} \mu_2.$$

The equivalence shows that approximate couplings can capture the bounds $\mu_1(\mathcal{S}_1) \leq \alpha\mu_2(\mathcal{S}_2) + \delta$ and $\mu_1(\mathcal{A}_1 \setminus \mathcal{S}_1) \leq \alpha\mu_2(\mathcal{A}_2 \setminus \mathcal{S}_2) + \delta$ with the most precise approximation parameters, much like the maximal coupling can precisely model the TV-distance between two distributions.

Proof. The reverse direction follows by Theorem 5.1.2. For the forward implication, take any set $\mathcal{T}_1 \subseteq \mathcal{A}_1$ and write $\mathcal{R}$ for the relation $\{(a_1, a_2) \mid a_1 \in \mathcal{S}_1 \leftrightarrow a_2 \in \mathcal{S}_2\}$. If $\mathcal{T}_1 \cap \mathcal{S}_1$ and $\mathcal{T}_1 \cap (\mathcal{A}_1 \setminus \mathcal{S}_1)$ are both non-empty, then $\mathcal{R}(\mathcal{T}_1) = \mathcal{A}_2$ and then clearly $\mu_1(\mathcal{T}_1) \leq \alpha\mu_2(\mathcal{R}(\mathcal{T}_1)) + \delta$ as μ_1 and μ_2 have equal weights. Otherwise $\mathcal{T}_1$ is contained in $\mathcal{S}_1$ or in $\mathcal{A}_1 \setminus \mathcal{S}_1$. In the first case, $\mathcal{R}(\mathcal{T}_1) = \mathcal{S}_2$ and so

$$\mu_1(\mathcal{T}_1) \leq \mu_1(\mathcal{S}_1) \leq \alpha\mu_2(\mathcal{S}_2) + \delta = \alpha\mu_2(\mathcal{R}(\mathcal{T}_1)) + \delta$$

by assumption. In the second case, $\mathcal{R}(\mathcal{T}_1) = \mathcal{A}_2 \setminus \mathcal{S}_2$ and we again have $\mu_1(\mathcal{T}_1) \leq \mu_2(\mathcal{R}(\mathcal{T}_1)) + \delta$. Hence we have the desired approximate coupling by Theorem 5.1.8.

Alternatively, we can directly construct two witnesses. For simplicity we consider just the case $\delta = 0$. Define:

$$\mu_L(a_1, a_2) \triangleq \begin{cases} \frac{\mu_1(a_1) \cdot \mu_2(a_2)}{\mu_2(\mathcal{S}_2)} & : \text{if } a_1 \in \mathcal{S}_1 \text{ and } a_2 \in \mathcal{S}_2 \\ \frac{\mu_1(a_1) \cdot \mu_2(a_2)}{\mu_2(\mathcal{A}_2 \setminus \mathcal{S}_2)} & : \text{if } a_1 \notin \mathcal{S}_1^\star \text{ and } a_2 \notin \mathcal{S}_2^\star \\ 0 & : \text{otherwise.} \end{cases}$$

$$\mu_R(a_1, a_2) \triangleq \begin{cases} \frac{\mu_1(a_1) \cdot \mu_2(a_2)}{\mu_1(\mathcal{S}_1)} & : \text{if } a_1 \in \mathcal{S}_1 \text{ and } a_2 \in \mathcal{S}_2 \\ \frac{\mu_1(a_1) \cdot \mu_2(a_2)}{\mu_1(\mathcal{A}_1 \setminus \mathcal{S}_1)} & : \text{if } a_1 \notin \mathcal{S}_1^\star \text{ and } a_2 \notin \mathcal{S}_2^\star \\ \mu_2(a_2) - \sum_{a_1' \in \mathcal{A}_1} \mu_R(a_1', a_2) & : \text{if } a_1 = \star \\ 0 & : \text{otherwise.} \end{cases}$$

When any denominator is zero, we treat the fraction as zero. It is not hard to see that the support conditions are satisfied. To show the marginal conditions, there are a few cases. Consider the first marginal $\pi_1(\mu_L)(a_1)$. For $a_1 \in \mathcal{S}_1$, if $\mu_2(\mathcal{S}_2) = 0$ then $\mu_1(\mathcal{S}_1) = 0$ by assumption; if $\mu_1(\mathcal{S}_2) > 0$ then the marginal is clear. Likewise for $a_1 \notin \mathcal{S}_1^\star$, if $\mu_2(\mathcal{A}_2 \setminus \mathcal{S}_2) = 0$ then $\mu_1(\mathcal{A}_1 \setminus \mathcal{S}_1) = 0$ by assumption and $\pi_1(\mu_L)(a_1) = 0$; if $\mu_2(\mathcal{A}_2 \setminus \mathcal{S}_2) > 0$ then the marginal is clear. The second marginal $\pi_2(\mu_R) = \mu_2$ holds by construction, after checking $\mu_R(\star, a_2) \geq 0$.

Finally for the distance condition, $\mu_L(a_1, a_2) \leq \alpha\mu_R(a_1, a_2)$ by the first assumption when $(a_1, a_2) \in \mathcal{S}_1 \times \mathcal{S}_2$; by the second assumption when $(a_1, a_2) \in (\mathcal{A}_1 \setminus \mathcal{S}_1) \times (\mathcal{A}_2 \setminus \mathcal{S}_2)$; and trivially in all other cases since $\mu_L(a_1, a_2) = \mu_R(a_1, a_2) = 0$. Hence we have a $(\ln \alpha, 0)$-approximate coupling. $\square$

A useful special case is when the distributions are equal and the subsets are nested.

Corollary 5.3.2 (Optimal subset coupling). *Let μ be a sub-distribution over $\mathcal{A}$ and consider nested sets $\mathcal{S}_2 \subseteq \mathcal{S}_1 \subseteq \mathcal{A}$. Then $\mu(\mathcal{S}_1) \leq \alpha\mu(\mathcal{S}_2) + \delta$ if and only if*

$$\mu \{(a_1, a_2) \mid a_1 \in \mathcal{S}_1 \leftrightarrow a_2 \in \mathcal{S}_2\}^{\sharp(\ln \alpha, \delta)} \mu.$$

Proof. By Theorem 5.3.1; the requirement $\mu(\mathcal{A} \setminus \mathcal{S}_1) \leq \alpha\mu(\mathcal{A} \setminus \mathcal{S}_2) + \delta$ is automatic since $\mathcal{S}_2 \subseteq \mathcal{S}_1$. $\square$

As an application, we give a subset coupling for the Laplace distribution. First, we prove a bound relating the probabilities of two nested intervals for the Laplace distribution. A similar bound for the continuous Laplace distribution was originally proved by Bun et al. (2017); we adapt their proof to the discrete case.

Proposition 5.3.3. *Let $a, a', b, b' \in \mathbb{Z}$ be such that $a < b$ and $[a, b] \subseteq [a', b']$. Then*

$$\Pr_{r \sim \mathbf{Lap}_\varepsilon} [r \in [a', b']] \leq \alpha \Pr_{r \sim \mathbf{Lap}_\varepsilon} [r \in [a, b]]$$

with constants

$$\alpha \triangleq \frac{\exp(\eta\varepsilon)}{1 - \exp(-(b - a + 2)\varepsilon/2)} \quad \text{and} \quad \eta \triangleq (b' - a') - (b - a).$$

Proof. Let W be the total mass of the Laplace distribution before normalization. By a calculation,

$$W = \sum_{r=-\infty}^{+\infty} \exp(-|r|\varepsilon) = \frac{e^\varepsilon + 1}{e^\varepsilon - 1}.$$

Let $L(x, y)$ be the mass of the Laplace distribution in $[x, y]$. We want to bound $L(a', b') \leq \alpha L(a, b)$. There are four cases: $a < b \leq 0$, $a < 0 < b$ with $|a| \leq |b|$, $0 \leq a < b$, and $a < 0 < b$ with $|a| \geq |b|$. By symmetry of the Laplace distribution, it suffices to consider the first two cases.

For the first case, $a < b \leq 0$. By direct calculation, we have

$$L(a', b') \leq L(a, b) + \frac{1}{W} \sum_{r=b+1}^{b+\eta} e^{r\varepsilon}$$

$$= \frac{e^{(b+1+\eta)\varepsilon} - e^{a\varepsilon}}{e^\varepsilon + 1}$$

$$= \frac{1}{e^\varepsilon + 1}(e^{(b+1)\varepsilon} - e^{a\varepsilon})\left(\frac{e^{\eta\varepsilon} - e^{-(b-a+1)\varepsilon}}{1 - e^{-(b-a+1)\varepsilon}}\right)$$

$$= \left(\frac{e^{\eta\varepsilon} - e^{-(b-a+1)\varepsilon}}{1 - e^{-(b-a+1)\varepsilon}}\right) L(a, b) \leq \alpha L(a, b).$$

For the second case, $a < 0 < b$ with $|a| \leq |b|$. We can bound

$$L(a', b') \leq L(a, b) + \eta L(a, a) = L(a, b) + \eta\left(\frac{e^\varepsilon - 1}{e^\varepsilon + 1}\right)e^{a\varepsilon}$$

$$\leq \left(1 + \eta\left(\frac{e^\varepsilon - 1}{e^\varepsilon + 1}\right)\frac{e^{a\varepsilon}}{L(0, b)}\right) L(a, b)$$

$$= \left(1 + \frac{\eta(e^\varepsilon - 1)e^{a\varepsilon}}{e^\varepsilon - e^{-b\varepsilon}}\right) L(a, b)$$

$$= \left(\frac{1 - e^{-(b+1)\varepsilon} + \eta(e^\varepsilon - 1)e^{(a-1)\varepsilon}}{1 - e^{-(b+1)\varepsilon}}\right) L(a, b)$$

$$\leq \left(\frac{1 + \eta(e^\varepsilon - 1)}{1 - e^{-(b+1)\varepsilon}}\right) L(a, b)$$

$$\leq \left(\frac{e^{2\eta\varepsilon}}{1 - e^{-(b-a+2)\varepsilon/2}}\right) L(a, b) \leq \alpha L(a, b).$$

$$\varepsilon' \triangleq \ln\left(\frac{\exp(\eta\varepsilon)}{1-\exp(-\sigma\varepsilon/2)}\right) \qquad x_1, x_2 \notin \mathrm{FV}(p,q,r,s,e_1,e_2)$$

$$\Phi \triangleq \begin{cases} |e_1\langle 1\rangle - e_2\langle 2\rangle| \le k \\ p+k \le r < s \le q-k \wedge (q-p)-(s-r) \le \eta \wedge 0 < \sigma \le (s-r)+2 \\ \forall w_1, w_2 \in \mathbb{Z}, \ (w_1 \in [p,q] \leftrightarrow w_2 \in [r,s]) \to \Psi\{w_1, w_2/x_1\langle 1\rangle, x_2\langle 2\rangle\} \end{cases}$$

$$\text{LAPINT} \ \frac{}{\vdash x_1 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_1) \sim_{(\varepsilon',0)} x_2 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_2) : \Phi \Longrightarrow \Psi}$$

Figure 5.4: Interval coupling rule $[\text{LAPINT}]$ for APRHL

The last line is because $(b+1) \ge (b-a+2)/2$, and because $1 + \eta(e^\varepsilon - 1) \le e^{\eta\varepsilon}$ for $\eta \in \mathbb{N}$ and $\varepsilon \ge 0$; to see this, note that equality holds at $\eta = 0$ and

$$\frac{1+(\eta+1)(e^\varepsilon - 1)}{1+\eta(e^\varepsilon - 1)} \le \frac{e^{(\eta+1)\varepsilon}}{e^{\eta\varepsilon}} = e^\varepsilon$$

for $\varepsilon \ge 0$, so the inequality is preserved as we increase η. $\square$

As a corollary, we have a subset coupling for the Laplace distribution.

Lemma 5.3.4. *Let $a, a', b, b' \in \mathbb{Z}$ be such that $a < b$ and $[a,b] \subseteq [a',b']$. We have an approximate lifting*

$$\mathbf{Lap}_\varepsilon \ \{(r_1, r_2) \mid r_1 \in [a',b'] \leftrightarrow r_2 \in [a,b]\}^{\sharp(\ln\alpha,0)} \ \mathbf{Lap}_\varepsilon$$

with constants

$$\alpha \triangleq \frac{\exp(\eta\varepsilon)}{1-\exp(-(b-a+2)\varepsilon/2)} \quad and \quad \eta \triangleq (b'-a') - (b-a).$$

Proof. Immediate by the forward direction of Corollary 5.3.2 and Proposition 5.3.3. $\square$

To use this coupling in APRHL, we introduce the rule $[\text{LAPINT}]$ in Fig. 5.4. To gain intuition, the following rule is a simplified special case:

$$\varepsilon' \triangleq \ln\left(\frac{\exp(\eta\varepsilon)}{1-\exp(-\sigma\varepsilon/2)}\right) \qquad x \notin \mathrm{FV}(p,q,r,s) \qquad \models \Phi \to \begin{cases} |e\langle 1\rangle - e\langle 2\rangle| \le k \\ p+k \le r < s \le q-k \\ (q-p)-(s-r) \le \eta \\ 0 < \sigma \le (s-r)+2 \end{cases}$$

$$\text{LAPINT}^* \ \frac{}{\vdash x \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e) \sim_{(\varepsilon',0)} x \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e) : \Phi \Longrightarrow x\langle 1\rangle \in [p,q] \leftrightarrow x\langle 2\rangle \in [r,s]}$$

Ignoring the technical side-conditions, this rule gives an approximate coupling relating the samples in $[p,q]$ in the first distribution with the samples in $[r,s]$ in the second distribution. The general rule $[\text{LAPINT}]$ can prove post-conditions of any shape.

Theorem 5.3.5. *The rule $[\text{LAPINT}]$ is sound.*

Proof. We leave the logical context ρ implicit. Let $V \triangleq \mathcal{X} \setminus \{x_1, x_2\}$ be the non-sampled variables; we write $m[V]$ for the restriction of a memory m to variables in V. Consider two memories m_1, m_2 and let the means $v_1 \triangleq [\![e_1]\!]m_1$ and $v_2 \triangleq [\![e_2]\!]m_2$ satisfy $|v_1 - v_2| \le k$. By the free variable condition, the expressions p, q, r, s are preserved by the

command so we will abuse notation and treat p, q, r, s as integer constants satisfying the pre-condition Φ. Let the output distributions be

$$\mu_1 \triangleq [\![x_1 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_1)]\!] m_1 \quad \text{and} \quad \mu_2 \triangleq [\![x_2 \xleftarrow{\$} \mathbf{Lap}_\varepsilon(e_2)]\!] m_2.$$

We construct an approximate coupling of μ_1 and μ_2. Define the intervals

$$\mathcal{I}_1 \triangleq [p - v_1, q - v_1] \quad \text{and} \quad \mathcal{I}_2 \triangleq [r - v_2, s - v_2].$$

Since $p + k \leq r$ and $s \leq q - k$ and $|v_1 - v_2| \leq k$, we know $\mathcal{I}_2 \subseteq \mathcal{I}_1$. Lemma 5.3.4 gives

$$\mathbf{Lap}_\varepsilon \ \{(r_1, r_2) \mid r_1 \in [p - v_1, q - v_1] \leftrightarrow r_2 \in [r - v_2, s - v_2]\}^{\sharp(\ln \alpha, 0)} \ \mathbf{Lap}_\varepsilon$$

with constants

$$\alpha \triangleq \frac{\exp(\eta \varepsilon)}{1 - \exp(-(s - r + 2)\varepsilon/2)} \quad \text{and} \quad \eta \triangleq (q - p) - (s - r).$$

Since $0 < \sigma \leq (s - r + 2)$, we have $\ln \alpha \leq \varepsilon'$ for

$$\varepsilon' \triangleq \ln\left(\frac{\exp(\eta \varepsilon)}{1 - \exp(-\sigma \varepsilon/2)} \right).$$

Proposition 5.3.3 yields an approximate coupling

$$\mathbf{Lap}_\varepsilon \ \{(r_1, r_2) \mid r_1 \in [p - v_1, q - v_1] \leftrightarrow r_2 \in [r - v_2, s - v_2]\}^{\sharp(\varepsilon', 0)} \ \mathbf{Lap}_\varepsilon.$$

Rearranging, this is equivalent to

$$\mathbf{Lap}_\varepsilon \ \{(r_1, r_2) \mid r_1 + v_1 \in [p, q] \leftrightarrow r_2 + v_2 \in [r, s]\}^{\sharp(\varepsilon', 0)} \ \mathbf{Lap}_\varepsilon.$$

Applying Theorem 4.2.7 with f_1, f_2 mapping r to $r + v_1$, $r + v_2$ respectively, we obtain

$$f_1^\sharp(\mathbf{Lap}_\varepsilon) \ \{(w_1, w_2) \mid w_1 \in [p, q] \leftrightarrow w_2 \in [r, s]\}^{\sharp(\varepsilon', 0)} \ f_2^\sharp(\mathbf{Lap}_\varepsilon).$$

Now since $f_1^\sharp(\mathbf{Lap}_\varepsilon) = \mathbf{Lap}_\varepsilon(v_1)$ and $f_2^\sharp(\mathbf{Lap}_\varepsilon) = \mathbf{Lap}_\varepsilon(v_2)$, we have

$$\mathbf{Lap}_\varepsilon(v_1) \ \{(w_1, w_2) \mid w_1 \in [p, q] \leftrightarrow w_2 \in [r, s]\}^{\sharp(\varepsilon', 0)} \ \mathbf{Lap}_\varepsilon(v_2).$$

Applying Theorem 4.2.7 with maps $[\![x_1]\!]$ and $[\![x_2]\!]$, we get

$$\mu_1 \ [\![x_1 \langle 1 \rangle \in [p, q] \leftrightarrow x_2 \langle 2 \rangle \in [r, s]]\!]^{\sharp(\varepsilon', 0)} \ \mu_2.$$

By the free variable condition, $m_1'[V] = m_1[V]$ and $m_2'[V] = m_2[V]$ for all memories $m_1' \in \mathrm{supp}(\mu_1)$ and $m_2' \in \mathrm{supp}(\mu_2)$, so we may assume by Proposition 4.2.6 that the witnesses are supported on such memories. Hence, we have witnesses to

$$\mu_1 \ \{(m_1', m_2') \mid m_1'[V] = m_1[V], \ m_2'[V] = m_2[V], \ m_1'(x_1) \in [p, q] \leftrightarrow m_2'(x_2) \in [r, s]\}^{\sharp(\varepsilon', 0)} \ \mu_2.$$

By the pre-condition, (m_1, m_2) satisfy

$$\forall w_1, w_2 \in \mathbb{Z}, \ w_1 \in [p, q] \leftrightarrow w_2 \in [r, s] \rightarrow \Psi \ \{w_1, w_2 / x_1 \langle 1 \rangle, x_2 \langle 2 \rangle\}$$

and so

$$\mu_1 \ \Psi^{\sharp(\varepsilon', 0)} \ \mu_2,$$

showing $[\textsc{LapInt}]$ is sound. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\square$

5.4 Advanced coupling composition

The sequencing rule [SEQ] in APRHL composes two approximate couplings while summing the approximation parameters; this rule is a generalization of the standard composition theorem of differential privacy (Theorem 4.1.4). In this section we extend the *advanced* composition theorem of differential privacy, Theorem 4.1.5, which allows trading off the ε and δ parameters when analyzing a composition of private mechanisms.

While the proof of the sequential composition theorem is fairly straightforward, the advanced composition theorem follows from a more technical argument using Azuma's inequality. It is not obvious how to extend the proof to approximate liftings, but fortunately we don't need to. The key observation is that the ε-distance condition on witnesses ensures differential privacy generalized to distributions over *pairs* of outputs. Therefore, we can directly generalize the advanced composition theorem to liftings by viewing the function mapping a pair of inputs to the left/right witness as itself differentially private.

However, there is an important catch: the advanced composition theorem assumes a *symmetric* adjacency relation. In particular, the witnesses must satisfy a two-sided, symmetric distance bound to compose, but approximate lifting only gives a one-sided bound for witnesses. So, we first introduce a symmetric version of approximate lifting where the witnesses satisfy the bound in both directions. Then we develop an advanced composition theorem for symmetric liftings in two stages. First we prove an advanced composition theorem for ε-distance, showing how to control the distance between the output distributions of two compositions if we can bound the symmetric distance between the output distributions of each step. Then, we give an advanced composition theorem given a symmetric approximate lifting at each step of a composition. To apply this principle in APRHL, we introduce a symmetric judgment in APRHL and show how to prove it from standard APRHL judgments, and we internalize advanced composition in a loop rule for symmetric judgments.

Remark 5.4.1. The advanced composition theorem from differential privacy implicitly assumes that all mechanisms terminate with probability 1, so in this section we assume all commands are lossless; this is not a serious restriction as derivable judgments in APRHL only relate lossless programs (Lemma 4.3.3).

Remark 5.4.2. While we focus on the advanced composition theorem, our technique provides a simple route to generalize other sequential composition theorems, like the optimal composition theorem and the heterogeneous composition theorem (Kairouz, Oh, and Viswanath, 2017), and composition theorems where the parameters can be selected adaptively (Rogers, Vadhan, Roth, and Ullman, 2016).

Symmetric approximate liftings

We first introduce a symmetric version of approximate lifting.

Definition 5.4.3. Let μ_1, μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$, and let $\mathcal{R} \subseteq \mathcal{A}_1 \times \mathcal{A}_2$ be a relation. Let $\star$ be an element disjoint from $\mathcal{A}_1$ and $\mathcal{A}_2$. Two sub-distributions μ_L, μ_R over pairs $\mathcal{A}_1^\star \times \mathcal{A}_2^\star$ are *witnesses* for the *symmetric* (ε, δ)-*approximate $\mathcal{R}$-lifting* of (μ_1, μ_2) if:

1. $\pi_1(\mu_L) = \mu_1$ and $\pi_2(\mu_R) = \mu_2$;

2. $\operatorname{supp}(\mu_L) \cup \operatorname{supp}(\mu_R) \subseteq \mathcal{R}^\star$; and

3. $d_\varepsilon(\mu_L, \mu_R) \leq \delta$ and $d_\varepsilon(\mu_R, \mu_L) \leq \delta$.

(Recall $\mathcal{S}^\star$ is the set $\mathcal{S} \cup \{\star\}$, and $\mathcal{R}^\star$ is the relation $\mathcal{R} \cup (\mathcal{A}_1 \times \{\star\}) \cup (\{\star\} \times \mathcal{A}_2)$.) When the particular witnesses are not important, we say μ_1 and μ_2 are related by the *symmetric* (ε, δ)-*lifting of $\mathcal{R}$*, denoted

$$\mu_1 \; \overline{\mathcal{R}}^{\sharp(\varepsilon, \delta)} \; \mu_2.$$

$\mathcal{R}$ need not be symmetric—in fact, $\mathcal{A}_1$ and $\mathcal{A}_2$ may be different sets.

This definition is nearly identical to standard approximate liftings (Definition 4.2.2) except it requires the distance bound in both directions. The two-sided bound in a symmetric lifting implies two standard approximate liftings: if $\mu_1 \ \overline{\mathcal{R}}^{\sharp(\varepsilon,\delta)} \ \mu_2$ holds, then $\mu_1 \ \mathcal{R}^{\sharp(\varepsilon,\delta)} \ \mu_2$ and $\mu_2 \ (\mathcal{R}^{-1})^{\sharp(\varepsilon,\delta)} \ \mu_1$ both hold by taking witnesses (μ_L, μ_R) and $(\mu_R^\top, \mu_L^\top)$ respectively, since $d_\varepsilon\left(\mu_R^\top, \mu_L^\top\right) = d_\varepsilon(\mu_R, \mu_L)$. In general, the converse may not be true. However when the relation $\mathcal{R}$ is of a particular form, we can construct a symmetric approximate lifting by giving two approximate liftings.

Lemma 5.4.4. *Suppose $\mathcal{S}_1, \mathcal{S}_2$ are subsets of $\mathcal{A}_1, \mathcal{A}_2$ respectively, and we have maps $f_1 : \mathcal{A}_1 \to \mathcal{B}$ and $f_2 : \mathcal{A}_2 \to \mathcal{B}$. Define a relation $\mathcal{R}$ on $\mathcal{A}_1 \times \mathcal{A}_2$ by*

$$a_1 \ \mathcal{R} \ a_2 \iff a_1 \in \mathcal{S}_1 \wedge a_2 \in \mathcal{S}_2 \wedge f_1(a_1) = f_2(a_2).$$

Let μ_1, μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$. The approximate liftings

$$\mu_1 \ \mathcal{R}^{\sharp(\varepsilon,\delta)} \ \mu_2 \qquad and \qquad \mu_2 \ (\mathcal{R}^{-1})^{\sharp(\varepsilon,\delta)} \ \mu_1,$$

imply the symmetric approximate lifting

$$\mu_1 \ \overline{\mathcal{R}}^{\sharp(\varepsilon,\delta)} \ \mu_2.$$

Proof. Let (μ_L, μ_R) witness $\mu_1 \ \mathcal{R}^{\sharp(\varepsilon,\delta)} \ \mu_2$ and let (ν_L, ν_R) witness $\mu_2 \ (\mathcal{R}^{-1})^{\sharp(\varepsilon,\delta)} \ \mu_1$. For every $b \in \mathcal{B}$, define subsets $[b]_{\mathcal{A}_1} \triangleq f_1^{-1}(b) \subseteq \mathcal{A}_1$ and $[b]_{\mathcal{A}_2} \triangleq f_2^{-1}(b) \subseteq \mathcal{A}_2$ partitioning $\mathcal{A}_1$ and $\mathcal{A}_2$. First, we have

$$
\begin{aligned}
\mu_1([b]_{\mathcal{A}_1}) &= \mu_L([b]_{\mathcal{A}_1} \times \mathcal{A}_2^\star) \\
&\leq \exp(\varepsilon) \cdot \mu_R([b]_{\mathcal{A}_1} \times \mathcal{A}_2^\star) + \delta \\
&= \exp(\varepsilon) \cdot \mu_R([b]_{\mathcal{A}_1} \times [b]_{\mathcal{A}_2}) + \delta \\
&\leq \exp(\varepsilon) \cdot \mu_R(\mathcal{A}_1^\star \times [b]_{\mathcal{A}_2}) + \delta \\
&= \exp(\varepsilon) \cdot \mu_2([b]_{\mathcal{A}_2}) + \delta.
\end{aligned}
$$

Define non-negative constants:

$$\rho(b) \triangleq \max(\mu_1([b]_{\mathcal{A}_1}) - \exp(\varepsilon) \cdot \mu_2([b]_{\mathcal{A}_2}), 0).$$

Then

$$\mu_1([b]_{\mathcal{A}_1}) \leq \exp(\varepsilon) \cdot \mu_2([b]_{\mathcal{A}_2}) + \rho(b),$$

with equality if $\rho(b) > 0$. It is not hard to show $\sum_{b \in \mathcal{B}} \rho(b) \leq \delta$; let $\mathcal{B}' \triangleq \{b \in \mathcal{B} \mid \rho(b) > 0\}$. Then

$$\mu_1(\cup_{b \in \mathcal{B}'} [b]_{\mathcal{A}_1}) = \exp(\varepsilon) \cdot \mu_2(\cup_{b \in \mathcal{B}'} [b]_{\mathcal{A}_2}) + \sum_{b \in \mathcal{B}'} \rho(b),$$

but Theorem 5.1.2 bounds the left side:

$$\mu_1(\cup_{b \in \mathcal{B}'} [b]_{\mathcal{A}_1}) \leq \exp(\varepsilon) \cdot \mu_2(\cup_{b \in \mathcal{B}'} [b]_{\mathcal{A}_2}) + \delta.$$

By a similar calculation with (ν_L, ν_R) in place of (μ_L, μ_R), we have a symmetric bound $\mu_2([b]_{\mathcal{A}_2}) \leq \exp(\varepsilon) \cdot \mu_1([b]_{\mathcal{A}_1}) + \sigma(b)$ for minimal non-negative constants $\sigma(b)$ such that $\sum_{b \in \mathcal{B}} \sigma(b) \leq \delta$. Note that $\rho(b)$ and $\sigma(b)$ can't both be strictly positive, by minimality. We define witnesses

$$
\eta_L(a_1, a_2) \triangleq
\begin{cases}
\frac{\mu_1(a_1) \cdot \mu_2(a_2)}{\mu_2([b]_{\mathcal{A}_2})} \left(1 - \frac{\rho(b)}{\mu_1([b]_{\mathcal{A}_1})}\right) & : f_1(a_1) = f_2(a_2) = b \\
\frac{\mu_1(a_1) \cdot \rho(b)}{\mu_1([b]_{\mathcal{A}_1})} & : a_2 = \star \\
0 & : \text{otherwise.}
\end{cases}
$$

$$\eta_R(a_1, a_2) \triangleq \begin{cases} \frac{\mu_1(a_1) \cdot \mu_2(a_2)}{\mu_1([b]_{\mathcal{A}_1})}\left(1 - \frac{\sigma(b)}{\mu_2([b]_{\mathcal{A}_2})}\right) & : f_1(a_1) = f_2(a_2) = b \\ \frac{\mu_2(a_2) \cdot \sigma(b)}{\mu_2([b]_{\mathcal{A}_2})} & : a_1 = \star \\ 0 & : \text{otherwise.} \end{cases}$$

Throughout, if a denominator is 0 we take the fraction to be 0 as well. Since $\text{supp}(\mu_1) \subseteq \mathcal{S}_1$ and $\text{supp}(\mu_2) \subseteq \mathcal{S}_2$ by the marginal and support conditions of the two asymmetric liftings, $\text{supp}(\eta_L)$ and $\text{supp}(\eta_R)$ are contained in $\mathcal{R}^\star$.

For the first marginal $\pi_1(\eta_L)(a_1)$, if $\mu_1([f_1(a_1)]_{\mathcal{A}_1})$ is zero then $\rho(f_1(a_1)) = 0$ by minimality and $\mu_1(a_1) = 0$, so $\eta_L(a_1, a_2) = 0$ for all $a_2 \in \mathcal{A}_2$. Otherwise if $\mu_2([f_1(a_1)]_{\mathcal{A}_2}) = 0$ then $\rho(f_1(a_1)) = \mu_1([f_1(a_1)]_{\mathcal{A}_1})$ by minimality, and $\eta_L(a_1, a_2) = \mu_1(a_1)$ for $a_2 = \star$ and zero for $a_2 \in \mathcal{A}_2$. By a symmetric argument, the second marginal is similar.

To check the symmetric distance conditions, take any set $\mathcal{W} \subseteq \mathcal{A}_1^\star \times \mathcal{A}_2^\star$. We want to compare

$$\eta_L(\mathcal{W}) = \sum_{(a_1, a_2) \in \mathcal{W}_0} \eta_L(a_1, a_2) + \sum_{(a_1, \star) \in \mathcal{W}} \eta_L(a_1, \star)$$

with

$$\eta_R(\mathcal{W}) = \sum_{(a_1, a_2) \in \mathcal{W}_0} \eta_R(a_1, a_2) + \sum_{(\star, a_2) \in \mathcal{W}} \eta_R(\star, a_2),$$

where $\mathcal{W}_0 \triangleq \mathcal{W} \cap (\mathcal{A}_1 \times \mathcal{A}_2)$. We claim (i) $\eta_L(a_1, a_2) \leq \exp(\varepsilon) \cdot \eta_R(a_1, a_2)$ for all $(a_1, a_2) \in \mathcal{A}_1 \times \mathcal{A}_2$, and (ii) $\sum_{(a_1, \star) \in \mathcal{W}} \eta_L(a_1, \star) \leq \delta$. Without loss of generality, we assume $\mathcal{W}$ is contained in $\mathcal{R}^\star$.

To show (i), let $b \triangleq f_1(a_1) = f_2(a_2)$. If either $\mu_1([b]_{\mathcal{A}_1})$ or $\mu_2([b]_{\mathcal{A}_2})$ are zero then the relevant probabilities in η_L and η_R are zero as well. Otherwise there are three cases. If both $\rho(b)$ and $\sigma(b)$ are both zero, then

$$\frac{\eta_L(a_1, a_2)}{\eta_R(a_1, a_2)} = \frac{\mu_1([b]_{\mathcal{A}_1})}{\mu_2([b]_{\mathcal{A}_2})} \leq \exp(\varepsilon).$$

If $\rho(b) > 0$, then $\sigma(b) = 0$ and $\mu_1([b]_{\mathcal{A}_1}) > 0$. If $\mu_2([b]_{\mathcal{A}_2}) = 0$ then the claim is immediate; otherwise,

$$\frac{\eta_L(a_1, a_2)}{\eta_R(a_1, a_2)} = \frac{\mu_1([b]_{\mathcal{A}_1})}{\mu_2([b]_{\mathcal{A}_2})}\left(1 - \frac{\rho(b)}{\mu_1([b]_{\mathcal{A}_1})}\right) = \frac{\mu_1([b]_{\mathcal{A}_1}) - \rho(b)}{\mu_2([b]_{\mathcal{A}_2})} = \exp(\varepsilon)$$

where the final equality is by minimality of $\rho(b)$. Similarly if $\sigma(b) > 0$, then $\rho(b) = 0$ and $\mu_2([b]_{\mathcal{A}_2}) > 0$ so

$$\frac{\eta_L(a_1, a_2)}{\eta_R(a_1, a_2)} = \frac{\mu_1([b]_{\mathcal{A}_1})}{\mu_2([b]_{\mathcal{A}_2})}\left(\frac{\mu_2([b]_{\mathcal{A}_2})}{\mu_2([b]_{\mathcal{A}_2}) - \sigma(b)}\right) = \frac{\mu_1([b]_{\mathcal{A}_1})}{\mu_2([b]_{\mathcal{A}_2}) - \sigma(b)} = \frac{\mu_1([b]_{\mathcal{A}_1})}{\exp(\varepsilon) \cdot \mu_1([b]_{\mathcal{A}_1})} \leq \exp(\varepsilon),$$

where the final equality is by minimality of $\sigma(b)$; note that if $\mu_2([b]_{\mathcal{A}_2}) = \sigma(b)$, then $\mu_1([b]_{\mathcal{A}_1}) = 0$, $\eta_L(a_1, a_2)$, and $\eta_R(a_1, a_2)$ are all zero. This establishes (i).

Showing (ii) is more straightforward:

$$\sum_{(a_1, \star) \in \mathcal{W}} \eta_L(a_1, \star) \leq \sum_{a_1 \in \mathcal{A}_1} \eta_L(a_1, \star) = \sum_{b \in \mathcal{B}} \rho(b) \leq \delta.$$

Hence we have

$$\begin{aligned} \eta_L(\mathcal{W}) &= \sum_{(a_1, a_2) \in \mathcal{W}_0} \eta_L(a_1, a_2) + \sum_{(a_1, \star) \in \mathcal{W}} \eta_L(a_1, \star) \\ &\leq \exp(\varepsilon) \sum_{(a_1, a_2) \in \mathcal{W}_0} \eta_R(a_1, a_2) + \delta \\ &\leq \exp(\varepsilon) \cdot \eta_R(\mathcal{W}) + \delta, \end{aligned}$$

giving the distance bound $d_\varepsilon(\eta_L, \eta_R) \leq \delta$. A similar calculation yields the symmetric bound $d_\varepsilon(\eta_R, \eta_L) \leq \delta$, so (η_L, η_R) witness the desired symmetric approximate lifting. $\square$

Advanced composition of symmetric ε-distance

Building up to advanced composition for symmetric approximate liftings, we first show advanced composition for symmetric ε-distance. Suppose we have two sequences of n functions $\{f_i\}_{i\in[n]}, \{g_i\}_{i\in[n]}$ where $f_i, g_i : \mathcal{A} \to \mathbf{Distr}(\mathcal{A})$ are such that for any $a \in \mathcal{A}$, we can bound the ε-distance between $f_i(a)$ and $g_i(a)$. Then we will bound the ε-distance between the output distributions from the n-fold compositions.

We use notation for the sequential composition of algorithms. Given a sequence of functions $\{h_i\}_{i\in[k]}$ where $h_i : \mathcal{A} \to \mathbf{Distr}(\mathcal{A})$, we write $h^k : \mathcal{A} \to \mathbf{Distr}(\mathcal{A})$ for the composition of $\{h_i\}$. Formally, we define

$$h^k(a) \triangleq \begin{cases} \mathbf{unit}(a) & : k = 0 \\ \mathbf{bind}(h^{k-1}(a), h_k) & : k > 0. \end{cases}$$

(Recall $\mathbf{unit} : \mathcal{A} \to \mathbf{Distr}(\mathcal{A})$ and $\mathbf{bind} : \mathbf{Distr}(\mathcal{A}) \times (\mathcal{A} \to \mathbf{Distr}(\mathcal{B})) \to \mathbf{Distr}(\mathcal{B})$ are the monadic operations for distributions from Definition 2.2.2.) We use the same notation for functions of type $h_i : \mathcal{D} \times \mathcal{A} \to \mathbf{Distr}(\mathcal{A})$, defining $h^k : \mathcal{D} \times \mathcal{A} \to \mathbf{Distr}(\mathcal{A})$ as

$$h^k(d, a) \triangleq \begin{cases} \mathbf{unit}(a) & : k = 0 \\ \mathbf{bind}(h^{k-1}(d, a), h_k(d, -)) & : k > 0. \end{cases}$$

Proposition 5.4.5. *Let $f_i, g_i : \mathcal{A} \to \mathbf{Distr}(\mathcal{A})$ satisfy $d_\varepsilon(f_i(a), g_i(a)) \le \delta$ and $d_\varepsilon(g_i(a), f_i(a)) \le \delta$ for every $i \in [n]$ and $a \in \mathcal{A}$. For any $\omega \in (0, 1)$, let*

$$\varepsilon^* \triangleq \varepsilon\sqrt{2n \ln(1/\omega)} + n\varepsilon(e^\varepsilon - 1) \quad and \quad \delta^* \triangleq n\delta + \omega.$$

Then for every $n \in \mathbb{N}$ and $a \in \mathcal{A}$, we have $d_{\varepsilon^*}(f^n(a), g^n(a)) \le \delta^*$ and $d_{\varepsilon^*}(g^n(a), f^n(a)) \le \delta^*$.

Proof. Let $\mathbb{B}$ be the booleans and define $h_i : \mathbb{B} \times \mathcal{A} \to \mathbf{Distr}(\mathcal{A})$ as

$$h_i(\mathbf{true}, a) \triangleq f_i(a) \quad and \quad h_i(\mathbf{false}, a) \triangleq g_i(a)$$

for every $a \in \mathcal{A}$. Then $d_\varepsilon(f_i(a), g_i(a)) \le \delta$ and $d_\varepsilon(g_i(a), f_i(a)) \le \delta$ imply $h_i(a, -) : \mathbb{B} \to \mathbf{Distr}(\mathcal{A})$ is (ε, δ)-differentially private for every $a \in \mathcal{A}$, where we view $\mathbb{B}$ as the set of databases with the full adjacency relation relating all pairs of booleans; in particular, this is a symmetric relation. Applying the advanced composition theorem of differential privacy (Theorem 4.1.5), $h^n(-, a) : \mathbb{B} \to \mathbf{Distr}(\mathcal{A})$ is $(\varepsilon^*, \delta^*)$-differentially private for every $a \in \mathcal{A}$. By Definition 4.2.1 we have

$$d_{\varepsilon^*}(h^n(\mathbf{true}, a), h^n(\mathbf{false}, a)) \le \delta^* \qquad \text{and} \qquad d_{\varepsilon^*}(h^n(\mathbf{false}, a), h^n(\mathbf{true}, a)) \le \delta^*$$

for every $a \in \mathcal{A}$. Since $h^n(\mathbf{true}, a) = f^n(a)$ and $h^n(\mathbf{false}, a) = g^n(a)$ by definition, we conclude

$$d_{\varepsilon^*}(f^n(a), g^n(a)) \le \delta^* \qquad \text{and} \qquad d_{\varepsilon^*}(g^n(a), f^n(a)) \le \delta^*. \qquad \square$$

Advanced composition of symmetric approximate liftings

Next, we extend Proposition 5.4.5 to symmetric approximate liftings; roughly speaking, we will apply the proposition to the functions mapping related inputs to the left or right witness distributions. We need a lemma about how witnesses are transformed under composition.

Lemma 5.4.6. *Consider two sequences of functions $\{f_i\}_{i\in[n]}, \{g_i\}_{i\in[n]}$ with $f_i : \mathcal{A}_1 \to \mathbf{Distr}(\mathcal{A}_1)$ and $g_i : \mathcal{A}_2 \to \mathbf{Distr}(\mathcal{A}_2)$, and a sequence of binary relations $\{\Phi_i\}_{i\in\{0,\dots,n\}}$ on $\mathcal{A}_1 \times \mathcal{A}_2$.*
Suppose we have two sequences of functions $\{l_i\}_{i\in[n]}, \{r_i\}_{i\in[n]}$ with $l_i, r_i : \mathcal{A}_1^\star \times \mathcal{A}_2^\star \to \mathbf{Distr}(\mathcal{A}_1^\star \times \mathcal{A}_2^\star)$ producing witnesses to an approximate lifting of Φ_i:

1. $\pi_1(l_i(a_1, a_2)) = f_i(a_1)$ and $\pi_2(r_i(a_1, a_2)) = g_i(a_2)$ for $(a_1, a_2) \in \Phi_{i-1}$;

2. $\pi_1(l_i(a_1, \star)) = f_i(a_1)$ and $\pi_2(r_i(\star, a_2)) = g_i(a_2)$; and

3. $supp(l_i(a_1, a_2)) \cup supp(r_i(a_1, a_2)) \subseteq \Phi_i^\star$ for $(a_1, a_2) \in \Phi_{i-1}^\star$

for every $i \in [n]$. Then l^n and r^n generate witnesses for an approximate lifting relating the n-fold compositions:

1. $\pi_1(l^n(a_1, a_2)) = f^n(a_1)$ and $\pi_2(r^n(a_1, a_2)) = g^n(a_2)$ for $(a_1, a_2) \in \Phi_0$;

2. $\pi_1(l^n(a_1, \star)) = f^n(a_1)$ and $\pi_2(r^n(\star, a_2)) = g^n(a_2)$; and

3. $supp(l^n(a_1, a_2)) \cup supp(r^n(a_1, a_2)) \subseteq \Phi_n^\star$ for every $(a_1, a_2) \in \Phi_0^\star$.

Proof. By induction on n. The base case $n = 0$ is trivial. When $n > 0$, the support condition follows by induction; the marginal conditions follow by a direct computation (Lemma A.1.1). $\square$

We are now ready to prove advanced composition for symmetric liftings.

Theorem 5.4.7. *Let $\omega \in (0, 1)$. Consider two sequences of functions $\{f_i\}_{i \in [n]}$ and $\{g_i\}_{i \in [n]}$ with $f_i : \mathcal{A}_1 \to$ **Distr**$(\mathcal{A}_1)$ and $g_i : \mathcal{A}_2 \to$ **Distr**$(\mathcal{A}_2)$, and a sequence of binary relations $\{\Phi_i\}_{i \in [n]}$ on $\mathcal{A}_1 \times \mathcal{A}_2$ and $\Phi_0 \subseteq \mathcal{A}_1 \times \mathcal{A}_2$. Suppose for every $i \in [n]$ and $(a_1, a_2) \in \Phi_{i-1}$, there is a symmetric approximate lifting:*

$$f_i(a_1) \; \overline{\Phi_i}^{\sharp(\varepsilon, \delta)} \; g_i(a_2).$$

Then for every $(a_1, a_2) \in \Phi_0$, we have a symmetric lifting

$$f^n(a_1) \; \overline{\Phi_n}^{\sharp(\varepsilon^*, \delta^*)} \; g^n(a_2)$$

where $\varepsilon^* \triangleq \varepsilon\sqrt{2n\ln(1/\omega)} + n\varepsilon(e^\varepsilon - 1)$ and $\delta^* \triangleq n\delta + \omega$.

Proof. For $(a_1, a_2) \in \Phi_{i-1}$, let $(\mu_L^{(i)}(a_1, a_2), \mu_R^{(i)}(a_1, a_2))$ witness the approximate lifting of Φ_i relating $f_i(a_1)$ and $g_i(a_2)$. Define functions $\{l_i\}_{i \in [n]}, \{r_i\}_{i \in [n]}$ of type $l_i, r_i : \mathcal{A}_1^\star \times \mathcal{A}_2^\star \to$ **Distr**$(\mathcal{A}_1^\star \times \mathcal{A}_2^\star)$ as follows:

$$l_i(a_1, a_2) \triangleq \begin{cases} \mu_L^{(i)}(a_1, a_2) & : (a_1, a_2) \in \Phi_{i-1} \\ \mathbf{unit}(\star) \times g_i(a_2) & : a_1 = \star, a_2 \neq \star \\ f_i(a_1) \times \mathbf{unit}(\star) & : a_1 \neq \star, a_2 = \star \\ \mathbf{unit}(\star, \star) & : a_1 = a_2 = \star \end{cases}$$

$$r_i(a_1, a_2) \triangleq \begin{cases} \mu_R^{(i)}(a_1, a_2) & : (a_1, a_2) \in \Phi_{i-1} \\ \mathbf{unit}(\star) \times g_i(a_2) & : a_1 = \star, a_2 \neq \star \\ f_i(a_1) \times \mathbf{unit}(\star) & : a_1 \neq \star, a_2 = \star \\ \mathbf{unit}(\star, \star) & : a_1 = a_2 = \star \end{cases}$$

Given distributions η_1 and η_2 over $\mathcal{B}_1$ and $\mathcal{B}_2$ respectively, $\eta_1 \times \eta_2 \in$ **Distr**$(\mathcal{B}_1 \times \mathcal{B}_2)$ denotes the product distribution defined in the expected way:

$$(\eta_1 \times \eta_2)(b_1, b_2) \triangleq \eta_1(b_1) \cdot \eta_2(b_2).$$

Now by assumption on $(\mu_L^{(i)}(a_1, a_2), \mu_R^{(i)}(a_1, a_2))$ and by definition when $a_1 = \star$ or $a_2 = \star$, we have

$$d_\varepsilon(l_i(a_1, a_2), r_i(a_1, a_2)) \leq \delta \qquad \text{and} \qquad d_\varepsilon(r_i(a_1, a_2), l_i(a_1, a_2)) \leq \delta$$

$$\Psi \triangleq e_1\langle 1\rangle = e_2\langle 2\rangle \wedge \Psi_1\langle 1\rangle \wedge \Psi_2\langle 2\rangle.$$

$$\text{SYMINTRO}\ \frac{\vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi \qquad \vdash c_2 \sim_{(\varepsilon,\delta)} c_1 : \Phi^{-1} \Longrightarrow \Psi^{-1}}{\vdash c_1 \approx_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi}$$

$$\text{SYMELIM-L}\ \frac{\vdash c_1 \approx_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi}{\vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi}$$

$$\text{SYMELIM-R}\ \frac{\vdash c_1 \approx_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi}{\vdash c_2 \sim_{(\varepsilon,\delta)} c_1 : \Phi^{-1} \Longrightarrow \Psi^{-1}}$$

Figure 5.5: Conversion rules between symmetric and standard judgments for APRHL

for all $(a_1, a_2) \in \Phi_{i-1}^\star$, and we have the marginal conditions required by Proposition 5.4.5. Now take any $(a_1, a_2) \in \Phi_0$. By Proposition 5.4.5, we have

$$d_{\varepsilon^*}(l^n(a_1,a_2), r^n(a_1,a_2)) \leq \delta^* \qquad \text{and} \qquad d_{\varepsilon^*}(r^n(a_1,a_2), l^n(a_1,a_2)) \leq \delta^*.$$

Lemma 5.4.6 gives the marginal conditions $\pi_1(l^n(a_1,a_2)) = f^n(a_1)$ and $\pi_2(r^n(a_1,a_2)) = g^n(a_2)$ and shows that $\text{supp}(l^n(a_1,a_2)), \text{supp}(r^n(a_1,a_2))$ are contained in $\Phi_n^\star$, so $l^n(a_1,a_2)$ and $r^n(a_1,a_2)$ witness the desired symmetric approximate lifting

$$f^n(a_1)\ \overline{\overline{\Phi_n}}^{\sharp(\varepsilon^*,\delta^*)}\ g^n(a_2). \qquad \qquad \square$$

Symmetric judgments in APRHL

In order to use advanced composition in APRHL, we extend the logic with a new judgment modeling symmetric approximate liftings. We call such judgments *symmetric* judgments.

Definition 5.4.8. A symmetric APRHL judgment is *valid* in logical context ρ, written

$$\rho \models c_1 \approx_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi,$$

if for any two inputs $(m_1, m_2) \in [\![\Phi]\!]_\rho$ there exists an symmetric approximate lifting relating the outputs:

$$[\![c_1]\!]_\rho m_1\ \overline{\overline{[\![\Psi]\!]_\rho}}^{\sharp([\![\varepsilon]\!]_\rho, [\![\delta]\!]_\rho)}\ [\![c_2]\!]_\rho m_2.$$

To prove these judgments, we extend APRHL with a few proof rules. To keep our proof system as simple as possible, we introduce rules for symmetric judgments only where absolutely needed—namely, for advanced composition—and use the conversion rules in Fig. 5.5 to move between symmetric and standard, asymmetric judgments. The inverse relation Φ^{-1} can be defined syntactically by simply interchanging the tags $\langle 1\rangle$ and $\langle 2\rangle$ in a formula Φ. Soundness of these rules is straightforward.

Theorem 5.4.9. *The rules* [SYMINTRO], [SYMELIM-L], *and* [SYMELIM-R] *are sound.*

Proof. Soundness of [SYMINTRO] follows by Lemma 5.4.4. Soundness of [SYMELIM-L] and [SYMELIM-R] follow by definition of symmetric approximate lifting. $\square$

$$\varepsilon^* \triangleq \varepsilon\sqrt{2N\ln(1/\omega)} + N\varepsilon(e^\varepsilon - 1) \qquad \delta^* \triangleq N\delta + \omega \qquad \omega \in (0, 1)$$

$$\text{WHILE-AC} \frac{\models \Phi \to e_\nu\langle 1\rangle \leq 0 \to \neg e_1\langle 1\rangle \qquad \models \Phi \to e_1\langle 1\rangle = e_2\langle 2\rangle}{\forall K \in \mathbb{N}, \vdash c_1 \approx_{(\varepsilon,\delta)} c_2 : \Phi \wedge e_1\langle 1\rangle \wedge e_\nu\langle 1\rangle = K \Longrightarrow \Phi \wedge e_\nu\langle 1\rangle < K}$$
$$\vdash \textbf{while } e_1 \textbf{ do } c_1 \approx_{(\varepsilon^*,\delta^*)} \textbf{while } e_2 \textbf{ do } c_2 : \Phi \wedge e_\nu\langle 1\rangle \leq N \Longrightarrow \Phi \wedge \neg e_1\langle 1\rangle$$

Figure 5.6: Advanced composition rule [WHILE-AC] for APRHL

An advanced composition rule for APRHL

Finally, we internalize advanced composition of liftings as the loop rule [WHILE-AC] in Fig. 5.6. Like the usual rule [WHILE], the guards must be synchronized and the loops run at most N iterations. An (ε, δ)-approximate coupling of the loop bodies gives an $(\varepsilon^*, \delta^*)$-approximate coupling of the two loops, where ε^* and δ^* are from the advanced composition theorem of differential privacy (Theorem 4.1.5).

Theorem 5.4.10. *The rule* [WHILE-AC] *is sound.*

Proof. The proof follows essentially by Theorem 5.4.7. As usual, we will leave the logical context ρ implicit. Consider two memories $(m_1, m_2) \in [\![\Phi \wedge e_\nu\langle 1\rangle \leq N]\!]$ and two output distributions

$$\mu_1 \triangleq [\![\textbf{while } e_1 \textbf{ do } c_1]\!]m_1 \quad \text{and} \quad \mu_2 \triangleq [\![\textbf{while } e_2 \textbf{ do } c_2]\!]m_2.$$

We construct a symmetric approximate lifting relating μ_1 and μ_2. The value of N is given by the logical context ρ; we treat it as a constant. We unroll the loop N times and define

$$\mu_1' \triangleq [\![(\textbf{if } e_1 \textbf{ then } c_1)^N]\!]m_1 \quad \text{and} \quad \mu_2' \triangleq [\![(\textbf{if } e_2 \textbf{ then } c_2)^N]\!]m_2.$$

We claim $[\![e_1]\!]m_1' = [\![e_2]\!]m_2' = \textbf{false}$ for all $m_1' \in \text{supp}(\mu_1')$ and $m_2' \in \text{supp}(\mu_2')$. We can use the valid symmetric APRHL judgment in the premise and symmetric versions of the rules [SEQ] and [COND] to construct a symmetric approximate lifting

$$\mu_1' \overline{\Phi \wedge e_\nu\langle 1\rangle \leq 0}^{\sharp(N\varepsilon, N\delta)} \mu_2'.$$

Since $\models \Phi \wedge e_\nu\langle 1\rangle \leq 0 \to \neg e_1\langle 1\rangle$, we have

$$\mu_1' \overline{\neg e_1\langle 1\rangle \wedge \neg e_2\langle 2\rangle}^{\sharp(N\varepsilon, N\delta)} \mu_2'.$$

Let μ_L', μ_R' be the corresponding witnesses. We know $\pi_1(\mu_L') = \mu_1'$ and $\pi_2(\mu_R') = \mu_2'$, and also

$$\text{supp}(\mu_L') \cup \text{supp}(\mu_R') \subseteq [\![\neg e_1\langle 1\rangle \wedge \neg e_2\langle 2\rangle]\!],$$

so $[\![e_1]\!]m_1' = [\![e_2]\!]m_2' = \textbf{false}$ for all m_1', m_2' in the support of μ_1', μ_2' respectively. By the equivalences

$$\textbf{while } e_1 \textbf{ do } c_1 \equiv (\textbf{if } e_1 \textbf{ then } c_1)^N; \textbf{while } e_1 \textbf{ do } c_1$$
$$\textbf{while } e_2 \textbf{ do } c_2 \equiv (\textbf{if } e_2 \textbf{ then } c_2)^N; \textbf{while } e_2 \textbf{ do } c_2,$$

we know

$$\mu_1 = [\![(\textbf{if } e_1 \textbf{ then } c_1)^N]\!]m_1 \quad \text{and} \quad \mu_2 = [\![(\textbf{if } e_2 \textbf{ then } c_2)^N]\!]m_2.$$

Defining a family of relations

$$\Phi_i \triangleq \Phi \wedge (e_\nu\langle 1\rangle \leq N - i \vee \neg e_1\langle 1\rangle),$$

we have

$$\models \textbf{if } e_1 \textbf{ then } c_1 \approx_{(\varepsilon,\delta)} \textbf{if } e_2 \textbf{ then } c_2 : \Phi_i \implies \Phi_{i+1}$$

for every i using the premise, since Φ_i ensures the guards e_1 and e_2 are equal in the initial memories. By validity, for any pair of memories satisfying Φ_i there is a symmetric approximate lifting of Φ_{i+1} relating the two output distributions. We can apply Theorem 5.4.7 with $\mathcal{A}_1 = \mathcal{A}_2 = \textbf{State}$, functions $f_i \triangleq [\![\textbf{if } e_1 \textbf{ then } c_1]\!]$ and $g_i \triangleq [\![\textbf{if } e_2 \textbf{ then } c_2]\!]$, and relations Φ_i to get the symmetric approximate lifting

$$\mu_1 \, \overline{\Phi \wedge (e_v\langle 1\rangle \leq 0 \vee \neg e_1\langle 1\rangle)}^{\sharp(\varepsilon^*,\delta^*)} \, \mu_2.$$

Since $\models \Phi \wedge e_v\langle 1\rangle \leq 0 \to \neg e_1\langle 1\rangle$, we conclude

$$\mu_1 \, \overline{\Phi \wedge \neg e_1\langle 1\rangle}^{\sharp(\varepsilon^*,\delta^*)} \, \mu_2$$

so [WHILE-AC] is sound.   □

Remark 5.4.11. Our approach narrowly limits the scope of symmetric judgments: they can be used in [WHILE-AC] or eliminated to a standard judgment. There are at least two other choices. One option would be to define a full proof system based on symmetric judgments. Almost all the basic proof rules from APRHL would directly generalize, including the standard rules for program commands and the Laplace rules. However, it is not clear how to generalize the more advanced rules, including [PW-EQ] and [UTB-L]/[UTB-R]. The optimal subset coupling (Theorem 5.3.1) also does not directly generalize to symmetric liftings; this poses a problem for a symmetric version of [LAPINT].

For another option, we could avoid symmetric judgments entirely by fusing [SYMINTRO], [WHILE-AC], and [SYMELIM-L] together into a single rule. While this would suffice for our examples, it is conceptually clearer to separate symmetric and asymmetric judgments. Our design choice leaves room for other rules specific to symmetric approximate liftings, and clearly identifies the main bottleneck in converting from standard approximate liftings to symmetric liftings in the rule [SYMINTRO].

5.5   Proving privacy for Between Thresholds

To draw everything together, we prove differential privacy for the *Between Thresholds* mechanism proposed by Bun et al. (2017), a more advanced version of the Sparse Vector mechanism. The input is again a stream of numeric queries, but now there are *two* numeric thresholds A and B with $A < B$. The original mechanism outputs LEFT if the answer is approximately below A, RIGHT if the answer is approximately above B, and HALT if the answer is approximately between A and B.

We analyze a variant of Between Thresholds that releases the index and approximate answer of the first C queries between the thresholds; Fig. 5.7 presents the code of the algorithm. The variables a and b contain the noisy thresholds. Unlike Sparse Vector, we resample the noise u when computing a and b after each between-threshold query—this is needed to analyze the outer loop by advanced composition. Also, the noise u is added in *opposite* directions to the two thresholds. Otherwise, the code is largely the same as Sparse Vector.

The privacy analysis of this algorithm is more complex than for Sparse Vector. First, privacy fails if the noisy thresholds a and b are too close together. Even if the exact thresholds A and B are far apart, there is always some small, non-zero probability that the noise u may be very large. Therefore the best we can hope for is (ε, δ)-differential privacy, where δ bounds the probability that the threshold noise is too large. Second, while the proof strategy for the inner loop remains broadly the same, in the critical iteration we must ensure that if one execution is between thresholds, then so is the other; we use the subset coupling for this purpose. Finally, we apply the advanced composition theorem to analyze the outer loop.

It will be useful to have a simpler bound on the approximation parameter for the subset coupling.

```
i ← 1;
out ← [];
while i ≤ N ∧ |out| < C do
    u ⟵$ Lap_ε'(0);
    a ← A − u;
    b ← B + u;
    go ← true;
    ans ← (0, 0);
    while i ≤ N ∧ go do
        v ⟵$ Lap_ε'/3(evalQ(i, d));
        if a < v < b then
            noisy ⟵$ Lap_ε'(evalQ(i, d));
            ans ← (i, noisy);
            out ← ans :: out;
            go ← false;
    i ← i + 1
```

Figure 5.7: Between Thresholds

Lemma 5.5.1. *Let* $\lambda \in (0, 1/2)$. *Suppose we have* $r, s \in \mathbb{Z}$ *such that*

$$s - r \geq \frac{6}{\lambda} \ln \frac{4}{\lambda} - 2,$$

and suppose we have two means $v_1, v_2 \in \mathbb{Z}$ *with* $|v_1 - v_2| \leq 1$. *Then we have an approximate lifting*

$$\mathbf{Lap}_{\lambda/3}(v_1) \{(x_1, x_2) \mid x_1 \in [r-1, s+1] \leftrightarrow x_2 \in [r, s]\}^{\sharp(\lambda, 0)} \mathbf{Lap}_{\lambda/3}(v_2).$$

Proof. By the soundness of [LAPINT] (Theorem 5.3.5), we have an approximate lifting

$$\mathbf{Lap}_{\lambda/3}(v_1) \{(x_1, x_2) \mid x_1 \in [r-1, s+1] \leftrightarrow x_2 \in [r, s]\}^{\sharp(\kappa, 0)} \mathbf{Lap}_{\lambda/3}(v_2)$$

where

$$\kappa \triangleq \ln\left(\frac{\exp(2\lambda/3)}{1 - \exp(-\sigma\lambda/6)}\right) \quad \text{and} \quad \sigma \triangleq (s - r) + 2.$$

We check $\kappa \leq \lambda$ assuming $\sigma \geq \frac{6}{\lambda} \ln \frac{4}{\lambda}$. Substituting, it suffices to show

$$\frac{\exp(2\lambda/3)}{1 - \lambda/4} \leq \exp(\lambda)$$

which is equivalent to

$$\lambda/4 + \exp(-\lambda/3) - 1 \leq 0.$$

Since the left side is convex in λ, the maximum occurs on the boundary of the domain. We can directly check the inequality at the endpoints $\lambda = \{0, 1/2\}$. □

We are now ready to prove privacy for Between Thresholds. As we did for Sparse Vector, we start with an informal proof by approximate coupling.

Theorem 5.5.2. *Let $\varepsilon, \delta \in (0,1)$ and let $q_1, \ldots, q_N : \mathcal{D} \to \mathbb{Z}$ be a list of 1-sensitive queries. If we set*

$$\varepsilon' \triangleq \frac{\varepsilon}{6\sqrt{2C \ln(2/\delta)}}$$

and the thresholds A, B are equal across both runs and satisfy

$$B - A \geq \frac{6}{\varepsilon'} \ln(4/\varepsilon') + \frac{2}{\varepsilon'} \ln(2/\delta C),$$

then the Between Thresholds algorithm (Fig. 5.7) is (ε, δ)-differentially private.

Proof by approximate coupling. Consider the outer loop body. We have $|u\langle 1\rangle| \leq (1/\varepsilon') \ln(2/\delta C)$ in the first process except with probability $\delta/2C$, and we couple $u\langle 1\rangle$ and $u\langle 2\rangle$ so $u\langle 1\rangle - 1 = u\langle 2\rangle$; this is an $(\varepsilon', \delta/2C)$-approximate coupling since the noise is drawn from $\mathbf{Lap}_{\varepsilon'}(0)$. The coupling ensures the noisy thresholds satisfy

$$a\langle 1\rangle + 1 = a\langle 2\rangle \quad \text{and} \quad b\langle 1\rangle = b\langle 2\rangle + 1. \tag{5.1}$$

Next, consider the inner loop. Each iteration, we approximately couple the processes so $ans\langle 1\rangle = ans\langle 2\rangle$. For any pair (j, y) with $j \in \mathbb{N}$ and $y \in \mathbb{Z}$, we construct an approximate coupling of the inner loops such that if ans on the first side is equal to (j, y), then so is ans on the second side; by pointwise equality, this will imply an approximate coupling with $ans\langle 1\rangle = ans\langle 2\rangle$.

As before, if $j \notin [1, N]$ the proof is trivial. Otherwise, we handle the inner iterations in one of two ways. On iterations $i \neq j$ we couple the samplings for v and $noisy$ with the null coupling, ensuring $|v\langle 1\rangle - v\langle 2\rangle| \leq 1$. This guarantees that before iteration j, if the first side is outside the thresholds, then so is the second side (by the coupling of the thresholds, Eq. (5.1)). We use $(0, 0)$-approximate couplings for these iterations.

On the critical iteration $i = j$, we use the optimal subset coupling when sampling v so that

$$v\langle 1\rangle \in [a\langle 1\rangle, b\langle 1\rangle] \longleftrightarrow v\langle 2\rangle \in [a\langle 2\rangle, b\langle 2\rangle]. \tag{5.2}$$

Given our accuracy bound on $|u\langle 1\rangle|$, the inner interval $[a\langle 2\rangle, b\langle 2\rangle]$ satisfies

$$b\langle 2\rangle - a\langle 2\rangle \geq \frac{6}{\varepsilon'} \ln(4/\varepsilon') - 2$$

under the threshold coupling, so Eq. (5.2) is an $(\varepsilon', 0)$-approximate coupling (Lemma 5.5.1). This coupling ensures the two processes behave the same at the conditional. If both processes are between thresholds, we apply the standard coupling for the Laplace mechanism so $noisy\langle 1\rangle = noisy\langle 2\rangle$; this is an $(\varepsilon', 0)$-approximate coupling. If both processes are not between thresholds then we don't sample $noisy$. So, we have an $(2\varepsilon', 0)$-approximate coupling for the inner loop such that if ans is equal to (j, y) on the first run, then ans is equal to (j, y) on the second run. By pointwise equality, this implies an $(2\varepsilon', 0)$-approximate coupling for the inner loop with $ans\langle 1\rangle = ans\langle 2\rangle$ as long as the threshold noises satisfy $u\langle 1\rangle - 1 = u\langle 2\rangle$ and the accuracy bound.

Combined with the initial $(\varepsilon', \delta/2C)$-approximate coupling for u, we have an $(2\varepsilon' + \varepsilon', \delta/2C + 0) = (3\varepsilon', \delta/2C)$-approximate coupling ensuring $ans\langle 1\rangle = ans\langle 2\rangle$ for the body of the outer loop. The outer loop executes at most C iterations, so by the advanced composition theorem (using the parameter setting from Footnote 2) we have an (ε, δ)-approximate coupling of the outer loops with $out\langle 1\rangle = out\langle 2\rangle$, establishing (ε, δ)-differential privacy. $\square$

We can give a more formal proof of privacy in APRHL. We work with the following, equivalent version of Between Thresholds:

$$
\begin{aligned}
&i \leftarrow 1; \\
&out \leftarrow []; \\
&\textbf{while } i \leq N \wedge |out| < C \textbf{ do} \\
&\quad u \xleftarrow{\$} \textbf{Lap}_{\varepsilon'}(0); \\
&\quad a \leftarrow A - u; \\
&\quad b \leftarrow B + u; \\
&\quad go \leftarrow \textbf{true}; \\
&\quad ans \leftarrow (0,0); \\
&\quad \textbf{while } i \leq N \textbf{ do} \\
&\qquad v \xleftarrow{\$} \textbf{Lap}_{\varepsilon'/3}(\textbf{evalQ}(i,d)); \\
&\qquad \textbf{if } a < v < b \wedge go \textbf{ then} \\
&\qquad\quad noisy \xleftarrow{\$} \textbf{Lap}_{\varepsilon'}(\textbf{evalQ}(i,d)); \\
&\qquad\quad ans \leftarrow (i, noisy); \\
&\qquad\quad go \leftarrow \textbf{false}; \\
&\qquad i \leftarrow i + 1; \\
&\quad \textbf{if } p_1(ans) \neq 0 \textbf{ then} \\
&\quad\quad i \leftarrow p_1(ans) + 1; \\
&\quad out \leftarrow ans :: out
\end{aligned}
$$

We call this program BT and the inner loop in. Compared to the algorithm in Fig. 5.7, the main difference is in the inner loop: each execution of in runs through all the queries, skipping the check once we have found a between-threshold query. More precisely, the flag go, which indicates we have not yet found a between-threshold query, is in the inner loop guard in Fig. 5.7 while it is in the between thresholds check in BT. After the inner loop, if a between-thresholds query was found then the index in ans must be non-zero, so the algorithm records the noisy answer and index, and resets the counter i to pick up after the last answered query. The inner loops in this version of the algorithm can be analyzed synchronously.

Theorem 5.5.3. *Let $\varepsilon, \delta \in (0, 1)$, let $q_1, \ldots, q_N : \mathcal{D} \to \mathbb{Z}$ be a list of 1-sensitive queries, and let the logical variables D_1, D_2 represent two adjacent databases. If we set*

$$
\varepsilon' \triangleq \frac{\varepsilon}{6\sqrt{2C \ln(2/\delta)}}
$$

in BT, and the thresholds A, B are equal across both runs and satisfy

$$
B - A \geq \frac{6}{\varepsilon'} \ln(4/\varepsilon') + \frac{2}{\varepsilon'} \ln(2/\delta C),
$$

then the following judgment holds:

$$
\vdash BT \sim_{(\varepsilon, \delta)} BT : d\langle 1 \rangle = D_1 \wedge d\langle 2 \rangle = D_2 \Longrightarrow out\langle 1 \rangle = out\langle 2 \rangle.
$$

Proof. The APRHL proof follows the approximate coupling proof in Theorem 5.5.2 closely. There are two main technicalities. First, we must take care to apply the rules that affect the parameter δ in the proper order. For instance, [PW-Eq] should be applied to pointwise judgments that are $(\varepsilon, 0)$-approximate couplings—if the pointwise judgment has $\delta > 0$, then [PW-Eq] will sum δ over all possible outputs. Since [LapAcc-L]/[LapAcc-R] and [While-AC] increase the δ parameter, we apply these rules below [PW-Eq] in the proof tree. Second, we need to make sure that the outer loop invariant is of the correct form so we can convert to a symmetric judgment and apply [While-AC].

At a high level, we apply [PW-EQ] on the inner loop assuming in the pre-condition that the threshold noise are coupled appropriately, and not too large. Then, we apply the accuracy bound [LAPACC-L] and threshold coupling [LAPGEN] for the first part of the outer loop body. Finally, we convert the standard APRHL judgment for the loop body to a symmetric judgment, applying [WHILE-AC] on the outer loop to conclude the proof.

Let's see this plan in action. We begin with the inner loop, *in*. We prove a pointwise judgment for the following, equivalent version of *in*, split into three stages:

$$
\begin{aligned}
&\textbf{while } i \leq N \wedge i < j \textbf{ do}\\
&\quad v \xleftarrow{\$} \textbf{Lap}_{\varepsilon'/3}(\textbf{evalQ}(i,d));\\
&\quad \textbf{if } a < v < b \wedge go \textbf{ then}\\
&\qquad noisy \xleftarrow{\$} \textbf{Lap}_{\varepsilon'}(\textbf{evalQ}(i,d));\\
&\qquad ans \leftarrow (i, noisy);\\
&\qquad go \leftarrow \textbf{false};\\
&\quad i \leftarrow i + 1;\\
&\textbf{while } i \leq N \wedge i = j \textbf{ do}\\
&\quad v \xleftarrow{\$} \textbf{Lap}_{\varepsilon'/3}(\textbf{evalQ}(i,d));\\
&\quad \textbf{if } a < v < b \wedge go \textbf{ then}\\
&\qquad noisy \xleftarrow{\$} \textbf{Lap}_{\varepsilon'}(\textbf{evalQ}(i,d));\\
&\qquad ans \leftarrow (i, noisy);\\
&\qquad go \leftarrow \textbf{false};\\
&\quad i \leftarrow i + 1;\\
&\textbf{while } i \leq N \textbf{ do}\\
&\quad v \xleftarrow{\$} \textbf{Lap}_{\varepsilon'/3}(\textbf{evalQ}(i,d));\\
&\quad \textbf{if } a < v < b \wedge go \textbf{ then}\\
&\qquad noisy \xleftarrow{\$} \textbf{Lap}_{\varepsilon'}(\textbf{evalQ}(i,d));\\
&\qquad ans \leftarrow (i, noisy);\\
&\qquad go \leftarrow \textbf{false};\\
&\quad i \leftarrow i + 1
\end{aligned}
$$

We call this program *in'*, the three loops $w_<$, $w_=$, and $w_>$, and the common loop body $body_{in}$. We implicitly maintain the invariant $d\langle 1 \rangle = D_1 \wedge d\langle 2 \rangle = D_2$ in all judgments and take the following global invariant:

$$
\Xi \triangleq \begin{cases} i\langle 1 \rangle = i\langle 2 \rangle \\ a\langle 1 \rangle + 1 = a\langle 2 \rangle \wedge b\langle 1 \rangle = b\langle 2 \rangle + 1 \wedge b\langle 2 \rangle - a\langle 2 \rangle \geq \frac{6}{\varepsilon'}\ln(4/\varepsilon') - 2 \\ [a\langle 1 \rangle, b\langle 1 \rangle] = [A - u\langle 1 \rangle, B + u\langle 1 \rangle] \wedge [a\langle 2 \rangle, b\langle 2 \rangle] = [A - u\langle 2 \rangle, B + u\langle 2 \rangle] \end{cases}
$$

Reading from top to bottom, this ensures (i) the loops are synchronized, (ii) the noisy thresholds are coupled and not too close, and (iii) the noisy thresholds share the noise u. Since *in'* does not modify the variables a, b and u, this assertion is preserved by the loops. Now let $(j, y) \in \mathbb{N} \times \mathbb{Z}$ be a possible value of *ans*. We define the following invariants for the three loops:

$$
\Theta_< \triangleq \Xi \wedge go\langle 1 \rangle \to go\langle 2 \rangle \wedge \neg(i\langle 1 \rangle \leq N \wedge i\langle 1 \rangle < j) \to i\langle 1 \rangle = j
$$

$$
\Theta_= \triangleq \Xi \wedge \begin{cases} go\langle 1 \rangle \to go\langle 2 \rangle \\ ans\langle 1 \rangle = (j, y) \to ans\langle 2 \rangle = (j, y) \\ \neg(i\langle 1 \rangle \leq N \wedge i\langle 1 \rangle = j) \to i\langle 1 \rangle = j + 1 \end{cases}
$$

$$
\Theta_> \triangleq \Xi \wedge i\langle 1 \rangle > j \wedge ans\langle 1 \rangle = (j, y) \to ans\langle 2 \rangle = (j, y)
$$

Now we proceed one loop at a time. First, we have

$$
\vdash body_{in} \sim_{(0,0)} body_{in} : \Theta_< \Longrightarrow \Theta_<
$$

by coupling the sampling for v with [LapNull] and using [Lap-L], [Lap-R], or [LapNull] to couple the samples for *noisy*. This ensures $|v\langle 1\rangle - v\langle 2\rangle| \le 1$; combined with the threshold coupling, we know that if the first side doesn't find a between-threshold query then neither does the second side, so $go\langle 1\rangle \to go\langle 2\rangle$. We get a coupling for the first loop by [While]:

$$\vdash w_< \sim_{(0,0)} w_< : \Theta_< \Longrightarrow \Theta_< \wedge \neg(i \le N \wedge i < j).$$

For the second loop, we prove

$$\vdash body_{in} \sim_{(2\varepsilon',0)} body_{in} : \Theta_= \Longrightarrow \Theta_=.$$

We couple the samplings for v with the subset coupling [LapInt], ensuring the two processes take the same path in the conditional. Since the thresholds are sufficiently apart (by Ξ) and the queries are 1-sensitive, [LapInt] is an $(\varepsilon', 0)$-approximate coupling by Lemma 5.5.1.

If both processes find between-threshold queries, then we couple the samplings for *noisy* with the standard Laplace rule [Lap] so $noisy\langle 1\rangle = noisy\langle 2\rangle$; this is an $(\varepsilon', 0)$-approximate coupling since the queries are 1-sensitive. Otherwise if both sides are outside the interval, we do not sample *noisy*. Thus, we have a $(2\varepsilon', 0)$-approximate coupling where if $ans\langle 1\rangle = (j, y)$, then $ans\langle 2\rangle = (j, y)$ too. Since the loop $w_=$ executes for exactly one iteration, [While] gives

$$\vdash w_= \sim_{(2\varepsilon',0)} w_= : \Theta_= \Longrightarrow \Theta_= \wedge \neg(i\langle 1\rangle \le N \wedge i\langle 1\rangle = j).$$

For the last loop we simply couple the samplings for v with the null coupling [LapNull] and use any zero-cost coupling for *noisy* ([Lap-L], [Lap-R], or [LapNull]), giving

$$\vdash w_> \sim_{(0,0)} w_> : \Theta_> \Longrightarrow \Theta_> \wedge \neg(i \le N).$$

Applying the rule of consequence with the implications

$$\models \Theta_< \wedge \neg(i\langle 1\rangle \le N \wedge i\langle 1\rangle < j) \to \Theta_=$$
$$\models \Theta_= \wedge \neg(i\langle 1\rangle \le N \wedge i\langle 1\rangle = j) \to \Theta_>,$$

we combine the loop judgments while summing the approximation parameters with [Seq] to get

$$\vdash in' \sim_{(2\varepsilon',0)} in' : \Xi \Longrightarrow ans\langle 1\rangle = (j, y) \to ans\langle 2\rangle = (j, y).$$

Pointwise equality [PW-Eq] completes the proof for the inner loop:

$$\vdash in' \sim_{(2\varepsilon',0)} in' : \Xi \Longrightarrow ans\langle 1\rangle = ans\langle 2\rangle.$$

Now let the outer loop by w_{out}, with body $body_{out}$. We ensure Ξ after the threshold samplings by applying [LapGen] and the accuracy bound [LapAcc-L], using an $(\varepsilon', \delta/2C)$-approximate coupling for the threshold samplings and showing

$$\vdash body_{out} \sim_{(3\varepsilon',\delta/2C)} body_{out} : (i, out)\langle 1\rangle = (i, out)\langle 2\rangle \Longrightarrow (i, out)\langle 1\rangle = (i, out)\langle 2\rangle.$$

Continuing to keep the adjacency condition $d_1\langle 1\rangle = D_1 \wedge d_2\langle 2\rangle = D_2$ implicit, we can apply [SymIntro] to get the symmetric judgment

$$\vdash body_{out} \approx_{(3\varepsilon',\delta/2C)} body_{out} : (i, out)\langle 1\rangle = (i, out)\langle 2\rangle \Longrightarrow (i, out)\langle 1\rangle = (i, out)\langle 2\rangle.$$

Taking the loop invariant $\Psi \triangleq (i, out)\langle 1\rangle = (i, out)\langle 2\rangle \wedge d\langle 1\rangle = D_1 \wedge d\langle 2\rangle = D_2$, the advanced composition rule [While-AC] gives

$$\vdash w_{out} \approx_{(\varepsilon,\delta)} w_{out} : \Psi \Longrightarrow \Psi$$

using the setting of ε' from Footnote 2. Converting back to a standard judgment by [SymElim-L] and handling the initial assignments, we conclude differential privacy:

$$\vdash BT \sim_{(\varepsilon,\delta)} BT : d\langle 1\rangle = D_1 \wedge d\langle 2\rangle = D_2 \Longrightarrow out\langle 1\rangle = out\langle 2\rangle.$$

$\square$

5.6 Comparison to other approximate liftings

The notion of approximate lifting has been formulated numerous times. We compare with several prior definitions in the discrete case. Research on the continuous case is ongoing; we summarize recent developments in the next chapter (Section 6.1).

Symmetric approximate liftings

While symmetric approximate liftings are less general than their asymmetric counterparts, they are interesting in their own right. In fact, our symmetric approximate liftings are equivalent to the approximate liftings proposed by Barthe et al. (2013c) in the original work on proving differential privacy via relational program logics. Unlike our definitions, which use two witnesses, their notion is based on a single witness.

Definition 5.6.1. Let μ_1, μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$, and let $\mathcal{R} \subseteq \mathcal{A}_1 \times \mathcal{A}_2$ be a relation. A sub-distribution μ over pair s$\mathcal{A}_1 \times \mathcal{A}_2$ is a *witness* for the *one-witness* (ε, δ)-*approximate $\mathcal{R}$-lifting* of (μ_1, μ_2) if:

1. $\pi_1(\mu) \leq \mu_1$ and $\pi_2(\mu) \leq \mu_2$;

2. $\mathrm{supp}(\mu) \subseteq \mathcal{R}$; and

3. $d_\varepsilon(\mu_1, \pi_1(\mu)) \leq \delta$ and $d_\varepsilon(\mu_2, \pi_2(\mu)) \leq \delta$.[1]

This definition is arguably closer to the spirit of probabilistic couplings: a single joint sub-distribution approximately modeling two given distributions as marginals.

Theorem 5.6.2. *Let μ_1, μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$, and let $\mathcal{R} \subseteq \mathcal{A}_1 \times \mathcal{A}_2$ be a relation. There is a one-witness (ε, δ)-approximate lifting of $\mathcal{R}$ in the sense of Definition 5.6.1 if and only if there is a symmetric approximate lifting*

$$\mu_1 \; \overline{\mathcal{R}}^{\sharp(\varepsilon, \delta)} \; \mu_2.$$

Proof. For the reverse direction, let (μ_L, μ_R) witness the symmetric approximate lifting and define $\eta \in \mathbf{SDistr}(\mathcal{A}_1 \times \mathcal{A}_2)$ as the pointwise minimum: $\eta(a_1, a_2) \triangleq \min(\mu_L(a_1, a_2), \mu_R(a_1, a_2))$. We check that η is a witness to an approximate lifting in the sense of Definition 5.6.1.

The support condition follows from the support condition for (μ_L, μ_R). The marginal conditions $\pi_1(\eta) \leq \mu_1$ and $\pi_2(\eta) \leq \mu_2$ also follow by the marginal conditions for (μ_L, μ_R). The only thing to check is the distance condition. Define non-negative constants

$$\delta(a_1, a_2) \triangleq \max(\mu_L(a_1, a_2) - \exp(\varepsilon) \cdot \mu_R(a_1, a_2), 0).$$

By the distance condition on (μ_L, μ_R),

$$\mu_L(a_1, a_2) \leq \exp(\varepsilon) \cdot \mu_R(a_1, a_2) + \delta(a_1, a_2)$$

with equality when $\delta(a_1, a_2) > 0$, and $\sum_{a_1, a_2} \delta(a_1, a_2) \leq \delta$. We claim

$$\min(\mu_L(a_1, a_2), \mu_R(a_1, a_2)) \geq \exp(-\varepsilon)(\mu_L(a_1, a_2) - \delta(a_1, a_2)).$$

If $\delta(a_1, a_2) = 0$ then $\mu_R(a_1, a_2) \geq \exp(-\varepsilon)\mu_L(a_1, a_2)$. Otherwise if $\delta(a_1, a_2) > 0$, then

$$\mu_R(a_1, a_2) = \exp(-\varepsilon)(\mu_L(a_1, a_2) - \delta(a_1, a_2)) \leq \mu_L(a_1, a_2)$$

---

[1] The original definition by Barthe et al. (2013c) involved a symmetric notion of ε-distance, and flipped the direction of both distances in this point. To keep notation uniform, we present their definition in terms of our (asymmetric) notion of ε-distance from Definition 4.2.1.

and the claim is again clear. Similarly, define

$$\delta'(a_1, a_2) \triangleq \max(\mu_R(a_1, a_2) - \exp(\varepsilon) \cdot \mu_L(a_1, a_2), 0).$$

We have

$$\mu_R(a_1, a_2) \le \exp(\varepsilon) \cdot \mu_L(a_1, a_2) + \delta'(a_1, a_2)$$

with equality when $\delta'(a_1, a_2) = 0$, and $\sum_{a_1, a_2} \delta'(a_1, a_2) \le \delta$. By analogous reasoning, we have

$$\min(\mu_L(a_1, a_2), \mu_R(a_1, a_2)) \ge \exp(-\varepsilon)(\mu_R(a_1, a_2) - \delta'(a_1, a_2)).$$

Now let $S_1 \subseteq A_1$ be any subset. Then:

$$
\begin{aligned}
\mu_1(S_1) - \exp(\varepsilon) \cdot \pi_1(\eta)(S_1) &= \sum_{a_1 \in S_1} \left(\mu_1(a_1) - \exp(\varepsilon) \sum_{a_2 \in A_2} \min(\mu_L(a_1, a_2), \mu_R(a_1, a_2)) \right) \\
&\le \sum_{a_1 \in S_1} \left(\mu_1(a_1) - \exp(\varepsilon) \sum_{a_2 \in A_2} \exp(-\varepsilon)(\mu_L(a_1, a_2) - \delta(a_1, a_2)) \right) \\
&= \sum_{a_1 \in S_1, a_2 \in A_2} \delta(a_1, a_2) \le \delta.
\end{aligned}
$$

The other marginal is similar: for any subset $S_2 \subseteq A_2$, we have

$$
\begin{aligned}
\mu_2(S_2) - \exp(\varepsilon) \cdot \pi_2(\eta)(S_2) &= \sum_{a_2 \in S_2} \left(\mu_2(a_2) - \exp(\varepsilon) \sum_{a_1 \in A_1} \min(\mu_L(a_1, a_2), \mu_R(a_1, a_2)) \right) \\
&\le \sum_{a_2 \in S_2} \left(\mu_2(a_2) - \exp(\varepsilon) \sum_{a_1 \in A_1} \exp(-\varepsilon)(\mu_R(a_1, a_2) - \delta'(a_1, a_2)) \right) \\
&= \sum_{a_2 \in S_2, a_1 \in A_1} \delta'(a_1, a_2) \le \delta.
\end{aligned}
$$

Thus, η witnesses the one-witness (ε, δ)-approximate lifting of $\mathcal{R}$.

The forward direction is more interesting. Let $\eta \in \textbf{SDistr}(A_1 \times A_2)$ be the single witness and define

$$
\begin{aligned}
\delta(a_1) &\triangleq \max(\mu_1(a_1) - \exp(\varepsilon) \cdot \pi_1(\eta)(a_1), 0) \\
\delta'(a_2) &\triangleq \max(\mu_2(a_2) - \exp(\varepsilon) \cdot \pi_2(\eta)(a_2), 0).
\end{aligned}
$$

By the distance conditions $d_\varepsilon(\mu_1, \pi_1(\eta)) \le \delta$ and $d_\varepsilon(\mu_2, \pi_2(\eta)) \le \delta$, we have $\delta(a_1), \delta'(a_2) \ge 0$ and

$$
\begin{aligned}
\mu_1(a_1) &\le \exp(\varepsilon) \cdot \pi_1(\eta)(a_1) + \delta(a_1) \\
\mu_2(a_2) &\le \exp(\varepsilon) \cdot \pi_2(\eta)(a_2) + \delta'(a_2),
\end{aligned}
$$

with equality when $\delta(a_1)$ or $\delta'(a_2)$ are strictly positive. Furthermore, $\sum_{a_1 \in A_1} \delta(a_1)$ and $\sum_{a_2 \in A_2} \delta'(a_2)$ are at most δ. Define witnesses $\mu_L, \mu_R \in \textbf{SDistr}(A_1^\star \times A_2^\star)$ as follows:

$$
\mu_L(a_1, a_2) \triangleq \begin{cases}
\eta(a_1, a_2) \cdot \frac{\mu_1(a_1) - \delta(a_1)}{\pi_1(\eta)(a_1)} & : a_1 \neq \star, a_2 \neq \star \\
\mu_1(a_1) - \sum_{a_2' \in A_2} \mu_L(a_1, a_2') & : a_1 \neq \star, a_2 = \star \\
0 & : \text{otherwise}
\end{cases}
$$

$$\mu_R(a_1, a_2) \triangleq \begin{cases} \eta(a_1, a_2) \cdot \frac{\mu_2(a_2) - \delta'(a_2)}{\pi_2(\eta)(a_2)} & : a_1 \neq \star, a_2 \neq \star \\ \mu_2(a_2) - \sum_{a_1' \in \mathcal{A}_1} \mu_R(a_1', a_2) & : a_1 = \star, a_2 \neq \star \\ 0 & : \text{otherwise.} \end{cases}$$

As usual, if any denominator is zero we take the whole term to be zero as well.

The support condition follows from the support condition of η; the marginal conditions hold by definition. All probabilities are non-negative—for instance in μ_L, if $\delta(a_1) > 0$ then $\mu_1(a_1) - \delta(a_1) = \exp(\varepsilon) \cdot \pi_1(\eta)(a_1) \geq 0$ and

$$\mu_L(a_1, \star) = \mu_1(a_1) - \exp(\varepsilon) \cdot \pi_1(\eta)(a_1) = \delta(a_1) \geq 0$$

when $\pi_1(\eta)(a_1) > 0$; if $\pi_1(\eta)(a_1) = 0$ then $\mu_L(a_1, \star) = \mu_1(a_1) = 0$. If $\delta(a_1) = 0$ then we can check $\eta(a_1, \star) \geq 0$. A similar argument shows that μ_R is non-negative.

So, it remains to check the distance bounds. We first claim

$$\mu_L(a_1, a_2) \leq \exp(\varepsilon) \cdot \mu_R(a_1, a_2) \qquad \text{and} \qquad \mu_R(a_1, a_2) \leq \exp(\varepsilon) \cdot \mu_L(a_1, a_2).$$

When $a_1, a_2 \neq \star$, by definition $\mu_L(a_1, a_2)$ and $\mu_R(a_1, a_2)$ are both positive or both zero depending on whether $\eta(a_1, a_2)$ is positive or zero. The zero case is immediate. In the positive case,

$$\frac{\mu_L(a_1, a_2)}{\eta(a_1, a_2)} = \frac{\mu_1(a_1) - \delta(a_1)}{\pi_1(\eta)(a_1)} \leq \exp(\varepsilon) \qquad \text{and} \qquad \frac{\mu_R(a_1, a_2)}{\eta(a_1, a_2)} = \frac{\mu_2(a_2) - \delta'(a_2)}{\pi_2(\eta)(a_2)} \leq \exp(\varepsilon).$$

We can also lower bound the ratios:

$$\frac{\mu_L(a_1, a_2)}{\eta(a_1, a_2)} = \frac{\mu_1(a_1) - \delta(a_1)}{\pi_1(\eta)(a_1)} \geq 1 \qquad \text{and} \qquad \frac{\mu_R(a_1, a_2)}{\eta(a_1, a_2)} = \frac{\mu_2(a_2) - \delta'(a_2)}{\pi_2(\eta)(a_2)} \geq 1;$$

for instance when $\delta(a_1) > 0$ the ratio is exactly equal to $\exp(\varepsilon) \geq 1$, and when $\delta(a_1) = 0$ the ratio is at least 1 by the marginal property $\pi_1(\eta) \leq \mu_1$. So, $\mu_L(a_1, a_2)/\eta(a_1, a_2)$ and $\mu_R(a_1, a_2)/\eta(a_1, a_2)$ are in $[1, \exp(\varepsilon)]$ when all distributions are positive, establishing the claim.

Finally, we bound the mass on points $(a_1, \star)$. Letting $\mathcal{S}_1 \subseteq \mathcal{A}_1$ be any subset, $0 = \mu_R(\mathcal{S}_1 \times \{\star\}) \leq \exp(\varepsilon) \cdot \mu_L(\mathcal{S}_1 \times \{\star\}) + \delta$ is clear. For the other direction,

$$\mu_L(\mathcal{S}_1 \times \{\star\}) = \sum_{a_1 \in \mathcal{S}_1} \left(\mu_1(a_1) - \mu_1(a_1) \sum_{a_2 \in \mathcal{A}_2} \frac{\eta(a_1, a_2)}{\pi_1(\eta)(a_1)} + \delta(a_1) \sum_{a_2 \in \mathcal{A}_2} \frac{\eta(a_1, a_2)}{\pi_1(\eta)(a_1)} \right)$$
$$= \mu_1(\mathcal{S}_1) - \mu_1(\mathcal{S}_1) + \delta(\mathcal{S}_1) \leq \exp(\varepsilon) \cdot \mu_R(\mathcal{S}_1 \times \{\star\}) + \delta.$$

The mass at points $(\star, a_2)$ can be bounded in a similar way. Let $\mathcal{S}_2 \subseteq \mathcal{A}_2$ be any subset. Then $0 = \mu_L(\{\star\} \times \mathcal{S}_2) \leq \exp(\varepsilon) \cdot \mu_R(\{\star\} \times \mathcal{S}_2) + \delta$ is clear. For the other direction,

$$\mu_R(\{\star\} \times \mathcal{S}_2) = \sum_{a_2 \in \mathcal{S}_2} \left(\mu_2(a_2) - \mu_2(a_2) \sum_{a_1 \in \mathcal{A}_1} \frac{\eta(a_1, a_2)}{\pi_2(\eta)(a_2)} + \delta'(a_2) \sum_{a_1 \in \mathcal{A}_1} \frac{\eta(a_1, a_2)}{\pi_2(\eta)(a_2)} \right)$$
$$= \mu_2(\mathcal{S}_2) - \mu_2(\mathcal{S}_2) + \delta'(\mathcal{S}_2) \leq \exp(\varepsilon) \cdot \mu_L(\{\star\} \times \mathcal{S}_2) + \delta.$$

So $d_\varepsilon(\mu_L, \mu_R) \leq \delta$ and $d_\varepsilon(\mu_R, \mu_L) \leq \delta$, and we have a symmetric approximate lifting. □

Asymmetric approximate liftings, alternative definition

After introducing their symmetric notion of lifting (Definition 5.6.1), Barthe et al. (2013c) also considered *asymmetric* approximate liftings with a single witness distribution.

Definition 5.6.3. Let μ_1, μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$, and let $\mathcal{R} \subseteq \mathcal{A}_1 \times \mathcal{A}_2$ be a relation. A sub-distribution μ over pairs $\mathcal{A}_1 \times \mathcal{A}_2$ is a *witness* for the *one-witness asymmetric (ε, δ)-approximate $\mathcal{R}$-lifting* of (μ_1, μ_2) if:

1. $\pi_1(\mu) \leq \mu_1$ and $\pi_2(\mu) \leq \mu_2$;

2. $\operatorname{supp}(\mu) \subseteq \mathcal{R}$; and

3. $d_\varepsilon(\mu_1, \pi_1(\mu)) \leq \delta$.[2]

Note the key difference compared to the symmetric version: the distance bound is only required to hold between the first distribution and the first marginal. We can show Definition 5.6.3 coincides with our asymmetric notion of approximate lifting.

Theorem 5.6.4. *Let μ_1, μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$, and let $\mathcal{R} \subseteq \mathcal{A}_1 \times \mathcal{A}_2$ be a relation. Then there is a (one-witness) asymmetric (ε, δ)-approximate lifting of $\mathcal{R}$ in the sense of Definition 5.6.3 if and only if there is an approximate lifting:*

$$\mu_1 \; \mathcal{R}^{\sharp(\varepsilon, \delta)} \; \mu_2.$$

Proof. For the reverse direction, let (μ_L, μ_R) witness the approximate lifting and define $\eta \in \mathbf{SDistr}(\mathcal{A}_1 \times \mathcal{A}_2)$ as the pointwise minimum: $\eta(a_1, a_2) \triangleq \min(\mu_L(a_1, a_2), \mu_R(a_1, a_2))$. We claim that η witnesses an asymmetric approximate lifting in the sense of Definition 5.6.3.

The support condition follows from the support condition for (μ_L, μ_R); the marginal conditions $\pi_1(\eta) \leq \mu_1$ and $\pi_2(\eta) \leq \mu_2$ also follow by the marginal conditions for (μ_L, μ_R). To check the distance condition, define

$$\delta(a_1, a_2) \triangleq \max(\mu_L(a_1, a_2) - \exp(\varepsilon) \cdot \mu_R(a_1, a_2), 0).$$

By the distance condition on (μ_L, μ_R), we have

$$\mu_L(a_1, a_2) \leq \exp(\varepsilon) \cdot \mu_R(a_1, a_2) + \delta(a_1, a_2)$$

with equality when $\delta(a_1, a_2) > 0$, and $\sum_{a_1, a_2} \delta(a_1, a_2) \leq \delta$. Like in the proof of Theorem 5.6.2, we have

$$\min(\mu_L(a_1, a_2), \mu_R(a_1, a_2)) \geq \exp(-\varepsilon)(\mu_L(a_1, a_2) - \delta(a_1, a_2)).$$

To conclude the distance bound, let $\mathcal{S}_1 \subseteq \mathcal{A}_1$ be a subset. Then:

$$\begin{aligned}
\mu_1(\mathcal{S}_1) - \exp(\varepsilon) \cdot \pi_1(\eta)(\mathcal{S}_1) &= \sum_{a_1 \in \mathcal{S}_1} \left(\mu_1(a_1) - \exp(\varepsilon) \sum_{a_2 \in \mathcal{A}_2} \min(\mu_L(a_1, a_2), \mu_R(a_1, a_2)) \right) \\
&\leq \sum_{a_1 \in \mathcal{S}_1} \left(\mu_1(a_1) - \exp(\varepsilon) \sum_{a_2 \in \mathcal{A}_2} \exp(-\varepsilon)(\mu_L(a_1, a_2) - \delta(a_1, a_2)) \right) \\
&= \sum_{a_1 \in \mathcal{S}_1, a_2 \in \mathcal{A}_2} \delta(a_1, a_2) \leq \delta.
\end{aligned}$$

Thus, η witnesses the (one-witness) asymmetric (ε, δ)-approximate lifting of $\mathcal{R}$.

The forward direction is more interesting. Let $\eta \in \mathbf{SDistr}(\mathcal{A}_1 \times \mathcal{A}_2)$ be the single witness and define

$$\delta(a_1) \triangleq \mu_1(a_1) - \exp(\varepsilon) \cdot \pi_1(\eta)(a_1).$$

---

[2] The original definition by Barthe et al. (2013c) used the same notion of ε-distance that we use (Definition 4.2.1), but incorrectly flipped the direction of the distance bound. It is also possible to define a version involving the second marginal instead of the first.

By the distance condition $d_\varepsilon(\mu_1, \pi_1(\eta)) \le \delta$, we know $\delta(a_1)$ is non-negative. Furthermore,

$$\mu_1(a_1) \le \exp(\varepsilon) \cdot \pi_1(\eta)(a_1) + \delta(a_1)$$

with equality when $\delta(a_1)$ is strictly positive, and $\sum_{a_1 \in \mathcal{A}_1} \delta(a_1) \le \delta$. Define two witnesses $\mu_L, \mu_R \in \mathbf{SDistr}(\mathcal{A}_1^\star \times \mathcal{A}_2^\star)$ as follows:

$$\mu_L(a_1, a_2) \triangleq \begin{cases} \eta(a_1, a_2) \cdot \frac{\mu_1(a_1) - \delta(a_1)}{\pi_1(\eta)(a_1)} & : a_1 \ne \star, a_2 \ne \star \\ \mu_1(a_1) - \sum_{a_2' \in \mathcal{A}_2} \mu_L(a_1, a_2') & : a_1 \ne \star, a_2 = \star \\ 0 & : \text{otherwise} \end{cases}$$

$$\mu_R(a_1, a_2) \triangleq \begin{cases} \eta(a_1, a_2) & : a_1 \ne \star, a_2 \ne \star \\ \mu_2(a_2) - \sum_{a_1' \in \mathcal{A}_1} \mu_R(a_1', a_2) & : a_1 = \star, a_2 \ne \star \\ 0 & : \text{otherwise.} \end{cases}$$

If any denominator is zero, we take the probability to be zero as well.

The support condition follows from the support condition of η; the marginal conditions hold by definition. To show all probabilities are non-negative, for μ_L note that if $\delta(a_1) > 0$ then $\mu_1(a_1) - \delta(a_1) = \exp(\varepsilon) \cdot \pi_1(\eta)(a_1) \ge 0$ and hence

$$\mu_L(a_1, \star) = \mu_1(a_1) - \delta(a_1) \ge 0$$

assuming $\pi_1(\eta)(a_1) > 0$; if $\pi_1(\eta)(a_1) = 0$ then $\mu_L(a_1, \star) = 0$. For μ_R, non-negativity holds by $\pi_2(\eta) \le \mu_2$.

We just need to show the distance bound. When $a_1, a_2 \ne \star$, we claim

$$\mu_L(a_1, a_2) \le \exp(\varepsilon) \cdot \eta(a_1, a_2) = \exp(\varepsilon) \cdot \mu_R(a_1, a_2).$$

By definition $\mu_L(a_1, a_2)$, $\mu_R(a_1, a_2)$, and $\eta(a_1, a_2)$ are all positive or all zero. The zero case is immediate. In the positive case,

$$\frac{\mu_L(a_1, a_2)}{\eta(a_1, a_2)} = \frac{\mu_1(a_1) - \delta(a_1)}{\pi_1(\eta)(a_1)} \le \exp(\varepsilon)$$

establishes the claim. To bound the mass on points $(a_1, \star)$, let $\mathcal{S}_1 \subseteq \mathcal{A}_1$ be any subset. Then:

$$\mu_L(\mathcal{S}_1 \times \{\star\}) = \sum_{a_1 \in \mathcal{S}_1} \left(\mu_1(a_1) - \mu_1(a_1) \sum_{a_2 \in \mathcal{A}_2} \frac{\eta(a_1, a_2)}{\pi_1(\eta)(a_1)} + \delta(a_1) \sum_{a_2 \in \mathcal{A}_2} \frac{\eta(a_1, a_2)}{\pi_1(\eta)(a_1)} \right)$$

$$= \mu_1(\mathcal{S}_1) - \mu_1(\mathcal{S}_1) + \delta(\mathcal{S}_1) \le \exp(\varepsilon) \cdot \mu_R(\mathcal{S}_1 \times \{\star\}) + \delta$$

so $d_\varepsilon(\mu_L, \mu_R) \le \delta$ as desired, and we have witnesses to an approximate lifting. □

Prior two-witness approximate liftings

Our notion of approximate lifting is strongly inspired by a prior definition.

Definition 5.6.5 (Barthe and Olmedo (2013) and Olmedo (2014))**.** Let μ_1, μ_2 be sub-distributions over $\mathcal{A}_1$ and $\mathcal{A}_2$, and let $\mathcal{R} \subseteq \mathcal{A}_1 \times \mathcal{A}_2$ be a relation. Two sub-distributions μ_L, μ_R over pairs $\mathcal{A}_1 \times \mathcal{A}_2$ are said to be *witnesses* for the (ε, δ)-*approximate $\mathcal{R}$-lifting* of (μ_1, μ_2) if:

1. $\pi_1(\mu_L) = \mu_1$ and $\pi_2(\mu_R) = \mu_2$;

2. $\mathrm{supp}(\mu_L) \cup \mathrm{supp}(\mu_R) \subseteq \mathcal{R}$; and

3. $d_\varepsilon(\mu_L, \mu_R) \le \delta$.

There are several positive features of this definition. First, it generalizes to other notions of distance on distribution; the distance d_ε can be replaced by an f-divergence. Furthermore, the witness distributions are related by a distance that looks like the distance from differential privacy, so composition theorems from differential privacy generalize to these liftings.

However, there are several notable drawbacks. Perhaps the biggest flaw is this definition does not support approximate lifting when $\mathcal{R}$ does not contain the supports $\mathrm{supp}(\mu_1) \times \mathrm{supp}(\mu_2)$. This limitation rules out up-to-bad couplings and accuracy bounds. There are also several annoying technical issues—the mapping property in Theorem 4.2.7 only holds for surjective maps, the support property Proposition 4.2.6 fails, the subset coupling in Theorem 5.3.1 does not work if the larger subset $\mathcal{S}_1$ is the whole domain $\mathcal{A}_1$, etc. These flaws are remedied in our definition.

Other notions of approximate equivalence

Approximate notions of lifting have also appeared in the literature on probabilistic bisimulation. Tschantz et al. (2011) introduced the δ-*lifting* of a relation $\mathcal{R}$ to relate two distributions μ_1, μ_2 when there is a bijection f on the supports matching elements with probabilities within a multiplicative factor:

$$\left| \ln \frac{\mu_1(x)}{\mu_2(f(x))} \right| \le \delta$$

and $(x, f(x)) \in \mathcal{R}$. Tschantz et al. (2011) used this notion of lifting to prove a variant of differential privacy for probabilistic labeled transition systems, with a proof technique based on an unwinding family of relations.

Prior researchers largely focused on additive notions of approximate equivalence; probably the first was due to Giacalone, Jou, and Smolka (1990). Segala and Turrini (2007) proposed ε-*lifting*, equivalent to $(0, \varepsilon)$-approximate lifting in our terminology. More recently, Desharnais, Laviolette, and Tracol (2008) and Tracol, Desharnais, and Zhioua (2011) investigated approximate notions of probabilistic simulation and bisimulation, again similar to our $(0, \delta)$-approximate liftings. Desharnais et al. (2008) noted the connection between their approximate liftings and maximum flows in a graph, extending the connection by Desharnais (1999, Theorem 7.3.4) for exact liftings; we use a similar observation to prove our approximate version of Strassen's theorem.

Chapter 6

Emerging directions

While we have limited this thesis to core connections between probabilistic couplings and program logics, several lines of work—recently completed or currently in progress—have already leveraged our results. We briefly survey these extensions (Section 6.1), and then discuss promising technical directions for further investigation (Section 6.2). We conclude by considering possible future connections between the theory of formal verification and the theory of randomized algorithms (Section 6.3).

6.1 Concurrent developments

Couplings for non-relational properties: Independence and uniformity

As we have seen, couplings are a natural fit for probabilistic relational properties. Properties describing a single program can also be viewed relationally in some cases, enabling cleaner proofs by coupling. Barthe, Espitau, Grégoire, Hsu, and Strub (2017b) develop this idea to prove *uniformity*, *probabilistic independence*, and *conditional independence*, examples of probabilistic non-relational properties. We briefly sketch their main reductions.

A uniform distribution places equal probability on every value in some range. Given a distribution μ over **State** and an expression e with finite range $\mathcal{S}$ (say, the booleans), e is *uniform* in μ if for all a and a' in $\mathcal{S}$, we have

$$\Pr_{m \sim \mu}[\llbracket e \rrbracket m = a] = \Pr_{m \sim \mu}[\llbracket e \rrbracket m = a'].$$

When μ is the output distribution of a program c, uniformity follows from the PRHL judgment

$$\forall a, a' \in \mathcal{S}, \vdash c \sim c : (=) \implies e\langle 1 \rangle = a \leftrightarrow e\langle 2 \rangle = a'.$$

This reduction is a direct consequence of Proposition 2.1.12. Moreover, the resulting judgment is ideally suited to relational verification since it relates two copies of the same program c.

Handling independence is only a bit more involved. Given a distribution μ and expressions e, e' with ranges $\mathcal{S}$ and $\mathcal{S}'$, we say e and e' are *probabilistically independent* if for all $a \in \mathcal{S}$ and $a' \in \mathcal{S}'$, we have

$$\Pr_{m \sim \mu}[\llbracket e \rrbracket m = a \wedge \llbracket e' \rrbracket m = a'] = \Pr_{m \sim \mu}[\llbracket e \rrbracket m = a] \cdot \Pr_{m \sim \mu}[\llbracket e' \rrbracket m = a'].$$

This useful property roughly implies that properties involving e and e' can be analyzed by focusing on e and e' separately. When e and e' are uniformly distributed, independence follows from uniformity of the tuple (e, e') over the product set $\mathcal{S} \times \mathcal{S}'$ so the previous reduction applies. In general, we can compare the distributions of e and e' in two experiments: when both are drawn from the output distribution of a single execution, and when they are

drawn from two independent executions composed sequentially. If the expressions are independent, these two experiments should look the same. Concretely, independence follows from the relational judgment

$$\forall a \in \mathcal{S}, a' \in \mathcal{S}', \vdash c \sim c^{(1)}; c^{(2)} : \Phi \implies e\langle 1 \rangle = a \wedge e'\langle 1 \rangle = a' \leftrightarrow e^{(1)}\langle 2 \rangle = a \wedge e'^{(2)}\langle 2 \rangle = a',$$

where $c^{(1)}$ and $c^{(2)}$ are copies of c with variables x renamed to $x^{(1)}$ and $x^{(2)}$ respectively; this construction is also called *self-composition* since it sequentially composes c with itself (Barthe et al., 2011b). The pre-condition Φ states that the three copies of each variable are initially equal: $x\langle 1 \rangle = x^{(1)}\langle 2 \rangle = x^{(2)}\langle 2 \rangle$. Handling conditional independence requires a slightly more complex encoding, but the general pattern remains the same: encode products of probabilities by self-composition and equalities by lifted equivalence $(\leftrightarrow)^{\sharp}$.

These reductions give a simple method to prove uniformity and independence. Other non-relational properties could benefit from a similar approach, especially in conjunction with more sophisticated program transformations in PRHL to relate different copies of the same sampling instruction.

Variable approximate couplings

As we saw in Chapters 4 and 5, approximate couplings are a powerful tool for proving differential privacy. To further enhance the proof technique, we can consider more precise ways of reasoning about the ε and δ parameters. To keep things simple, APRHL opts for the most straightforward approach: ε and δ are constants or logical variables, independent of the program state. This choice is reflected in the form and interpretation of the judgments:

$$c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \implies \Psi,$$

where ε and δ are treated as as mathematical constants. This approach supports clean composition—we can simply add up ε and δ parameters without regard to which variables are changed by the program—but it can be more convenient to think of ε and δ as depending on the current state. For example, we may want to assert $\varepsilon \leq n$ for a program variable n, representing some kind of counter.

However, it is not immediately clear what a state-dependent privacy parameter should mean, especially when the state is randomized. To give a suitable interpretation, we can look to the notion of a *privacy loss random variable* from the privacy literature. Roughly, the privacy parameter ε may be viewed as a function mapping outputs to costs:

$$\Pr_{x \sim \mu_1}[x = \xi] \leq \exp(\varepsilon(\xi)) \cdot \Pr_{x \sim \mu_2}[x = \xi]$$

for every ξ in the support of μ_1 and μ_2. Then, μ_1 induces a distribution $\varepsilon^{\sharp}(\mu_1)$ over privacy costs. If every cost in the support of this distribution is bounded by a constant ε_0, the output distributions μ_1, μ_2 satisfy the condition required for ε_0-differential privacy. (See the textbook by Dwork and Roth (2014) for a more thorough exposition.)

Albarghouthi and Hsu (2018) take inspiration from this idea and define an extension of approximate couplings called *variable approximate couplings*. Unlike approximate couplings, which require a distance bound between witnesses that is constant in ε over all pairs of samples, a variable approximate couplings allows ε to vary:

$$\forall (a_1, a_2) \in \mathcal{A}_1 \times \mathcal{A}_2, \ \mu_L(a_1, a_2) \leq \exp(\varepsilon(a_1, a_2)) \cdot \mu_R(a_1, a_2)$$

where $\varepsilon : \mathcal{A}_1 \times \mathcal{A}_2 \to \mathbb{R}$ is now a function. The result is a refinement of $(\varepsilon, 0)$-approximate coupling supporting a more precise, randomized notion of privacy cost.

We can broadly compare reasoning in terms of variable approximate couplings with reasoning in terms of approximate couplings (e.g., using a system like APRHL). The main difficulty with variable approximate couplings is analyzing sequential composition: now that each coupling has multiple costs associated with different samples, the cost after composing couplings may become quite complicated—we can't simply add the costs together. Furthermore, it isn't clear how to handle the additive parameter δ for proving (ε, δ)-privacy. At the same time, variable approximate couplings allow intuitive reasoning closer to the cost-based interpretation of privacy, where

the privacy level ε is regarded as a dynamic, possibly randomized quantity that accumulates as the program executes. Rather than bounding the cost by a constant at each stage of composition, we only need to bound the cost at the end of the computation; this flexibility can support significantly simpler proofs.

Albarghouthi and Hsu (2018) use these richer couplings to support fully automated proofs of $(\varepsilon, 0)$-differential privacy for challenging examples, including the Report-noisy-max and Sparse Vector mechanisms we saw in Chapter 4. Roughly speaking, they encode valid approximate coupling proofs with standard Horn clauses and a new kind of coupling constraint, and then solve the constraint systems with automated program verification and synthesis techniques. Variable approximate couplings simplify their proofs in two ways. First, by allow the privacy cost to be randomized during the analysis, there is no need to separate deterministic and randomized parts of the state. Second, their proofs can leverage more sophisticated approximate couplings like the variable version of the choice coupling from Section 4.8, making their invariants easier to discover automatically.

Expectation couplings

Probabilistic couplings and approximate couplings relate distributions over plain sets with no additional structure. Many sets come with a notion of distance, like the Euclidean distance on real vectors or the Hamming distance on finite sets. If $\mathfrak{d} : \mathcal{A} \times \mathcal{A} \to \mathbb{R}^+$ is a distance function on a set $\mathcal{A}$, the *Kantorovich distance* on distributions $\mathbf{Distr}(\mathcal{A})$ is defined as

$$\mathfrak{d}^{\sharp}(\mu_1, \mu_2) \triangleq \min_{\mu \in \Omega(\mu_1, \mu_2)} \mathbb{E}_{(a_1, a_2) \sim \mu} [\mathfrak{d}(a_1, a_2)],$$

where the minimum is taken over all couplings μ of (μ_1, μ_2). This is a well-studied notion in probability theory and the theory of optimal transport, increasingly seeing applications in computer science and beyond (e.g., Desharnais, Gupta, Jagadeesan, and Panangaden (2004); van Breugel and Worrell (2001a,b) consider logical aspects, Deng and Du (2009) survey applications in computer science, and Villani (2008) explores the mathematical theory). Intuitively, the Kantorovich distance lifts a distance $\mathfrak{d}$ on the ground set to a distance $\mathfrak{d}^{\sharp}$ on distributions, much like how probabilistic liftings lift a relation $\mathcal{R}$ on the ground set to a relation $\mathcal{R}^{\sharp}$ on distributions. Varying the ground distance recovers common distances on distributions as special cases.

Barthe, Espitau, Grégoire, Hsu, and Strub (2018) use the Kantorovich distance to define *expectation coupling*, a quantitative extension of probabilistic coupling. Given two distributions μ_1 and μ_2 on a set $\mathcal{A}$ equipped with a distance $\mathfrak{d}$, a coupling μ is a $(\mathfrak{d}, \delta)$-expectation coupling if the expected value of $\mathfrak{d}$ on μ is at most δ. To construct and reason about these couplings, Barthe et al. (2018) develop a relational program logic $\mathbb{E}$PRHL by augmenting the pre- and post-conditions in PRHL judgments with *pre-* and *post-distances*:

$$c_1 \sim_f c_2 : \{\Phi; \mathfrak{d}\} \Longrightarrow \{\Psi; \mathfrak{d}'\}.$$

The function $f : \mathbb{R} \to \mathbb{R}$ describes how the lifted post-distance can be bounded as a function of the pre-distance. Judgments are valid when for any two input memories (m_1, m_2) satisfying the pre-condition Φ, there is an $(\mathfrak{d}', f(\mathfrak{d}(m_1, m_2)))$-expectation coupling of the two output distributions with support in Ψ. Intuitively, valid judgments model Lipschitz-continuity or sensitivity, where the distance on input memories is $\mathfrak{d}$ and the distance on output distributions is the Kantorovich distance $\mathfrak{d}'^{\sharp}$.

$\mathbb{E}$PRHL judgments can be combined in various ways, reflecting the clean compositional properties of expectation couplings. For instance, when f is a non-decreasing affine function (i.e., $f(z) = \alpha \cdot z + \beta$ with $\alpha, \beta \geq 0$), judgments compose sequentially:

$$\text{SEQ} \frac{\vdash c_1 \sim_f c_2 : \{\Phi; \mathfrak{d}\} \Longrightarrow \{\Psi; \mathfrak{d}'\} \qquad \vdash c_1' \sim_{f'} c_2' : \{\Psi; \mathfrak{d}'\} \Longrightarrow \{\Theta; \mathfrak{d}''\}}{\vdash c_1; c_1' \sim_{f' \circ f} c_2; c_2' : \{\Phi; \mathfrak{d}\} \Longrightarrow \{\Theta; \mathfrak{d}''\}}$$

The transitivity rule, which combines two judgments relating $c_1 \sim c_2$ and $c_2 \sim c_3$ into a judgment relating $c_1 \sim c_3$, fully internalizes the path coupling principle (Bubley and Dyer, 1997) we saw in Chapter 3.

$\mathbb{E}$PRHL is particularly useful for proving quantitative relational properties. In PRHL, as we noted in Section 2.3, there is no way to reason about the probability of an event in the coupling. Our logic $\times$PRHL from Chapter 3 makes the coupling more explicit, but the logic can only construct the product program, not reason about it. In contrast, $\mathbb{E}$PRHL judgments can directly express quantitative properties of the coupling with the pre- and post-distances.

To demonstrate, Barthe et al. (2018) use $\mathbb{E}$PRHL to verify convergence for a Markov chain from population dynamics, and for the Glauber dynamics. In contrast to our proof from Section 3.6, which required reasoning about the product program and applying path coupling externally, the $\mathbb{E}$PRHL proof can be carried out almost entirely within the logic. Adding to the properties that can be handled, $\mathbb{E}$PRHL can also verify that the Stochastic Gradient Descent algorithm is *uniformly stable,* a quantitative property comparing a learning algorithm's expected error on two training sets (Bousquet and Elisseeff, 2002); this recently-proposed property is rapidly gaining currency in the machine learning community as a way to prevent overfitting (Hardt, Recht, and Singer, 2016).

Couplings in the continuous case

To simplify our presentation, in this thesis we have focused on discrete distributions. However, programs sampling from continuous distributions are quite common in the algorithms literature; many private algorithms, for instance, use samples from real-valued distributions like the Gaussian distribution and the standard Laplace distribution. Though most of our results should carry over, the continuous case introduces additional measure-theoretic technicalities. Designing a verification system supporting continuous distributions—say, a program logic where programs can sample from the Gaussian distribution—requires carefully handling these details.

While research historically evolved from exact liftings in PRHL to approximate liftings in APRHL, current work on the continuous case has jumped directly to approximate liftings. As we discussed in Section 5.1, Sato (2016) introduced a novel definition of approximate lifting without witness distributions in the continuous case, developing a continuous version of APRHL. Sato derived his approximate lifting using a categorical construction called *codensity lifting of monads* (also called $\top\top$-lifting), proposed by Katsumata and Sato (2015). Roughly speaking, this operation turns a monad on a base category D into a (possibly *indexed* or *graded*) monad on another category C, along a functor $C \to D$. This approach gives a highly generic way to lift monads to new categories, abstracting away many details about the specific categories. Codensity lifting also gives a more principled construction in some sense, as the lifting satisfies certain universal properties. However, the high level of abstraction can make it difficult to construct and manipulate these liftings; the current, clean form of Sato's lifting is followed significant simplifications after applying codensity lifting.

More recent work generalizes witness-based approximate liftings to the continuous case, giving an alternative, more flexible construction of approximate liftings that is easier to work with. Sato, Barthe, Gaboradi, Hsu, and Katsumata (2017) introduce *span-based liftings*, generalizing binary relations to categorical spans and supporting a broad class of divergences beyond ε-distance with good composition properties. Roughly speaking, maps between spans carry additional information needed for smooth composition in the continuous case. Sato and his collaborators develop span-based liftings and a relational program logic to verify differential privacy and various relaxations, including *Rényi differential privacy* (Mironov, 2017) and *zero-concentrated differential privacy* (Bun and Steinke, 2016). When specialized to ε-distance, span-based liftings are equivalent to Sato's witness-free liftings, giving an approximate version of Strassen's theorem in the continuous case.[1]

6.2 Promising directions

We envision further investigation along three broad axes: extending the theory, exploring new applications, and automating the proof technique.

---

[1]Tetsuya Sato, personal communication.

Theoretical directions

While the theory of probabilistic couplings has been well-developed in mathematics, our work suggests several natural directions for further theoretical study.

Defining approximate couplings.   Our definition of approximate lifting satisfies many clean theoretical properties, but it is not yet clear whether we have arrived at the right definition. More evidence is needed, possibly in the form of other natural properties satisfied by approximate liftings, equivalences with other well-established notions, or logical and categorical characterizations of approximate coupling; analogous results for probabilistic liftings may provide a useful guide (Desharnais et al., 2002, 2003; Fijalkow et al., 2017; Larsen and Skou, 1991).

Furthermore, Barthe and Olmedo (2013) and Olmedo (2014) consider approximate liftings where the differential privacy distance $d_\varepsilon(\mu_1, \mu_2)$ is generalized to any f-*divergence*, a broader class of distance-like measures between distributions. While it is straightforward to adapt our approximate liftings to f-divergences, there is currently little evidence this yields a good definition; for instance, a universal version of approximate lifting (similar to Sato's definition) for f-divergences is not known.

Completeness of the proof systems.   While the proof systems of ×PRHL and APRHL are sound, we did not establish completeness: valid judgments should be provable by applying the rules. Much like standard Hoare logic, the best we can hope for is *relative* completeness. Assuming an oracle for formulas in the assertion logic, can the proof system prove all valid judgments?

On this fundamental question, very little is known. For ×PRHL, relative completeness of standard Hoare logic combined with some basic program transformations give relative completeness for terminating, deterministic programs. However, the rules for random sampling are likely to be highly incomplete; for instance, there are many couplings beyond bijection couplings. Furthermore, there may be more fundamental obstacles to relative completeness: Kumar and Ramesh (2001) give an example of a Markov chain that is rapidly mixing but where no *causal* coupling can establish this fact; all couplings encoded by ×PRHL are causal couplings. This negative result doesn't directly rule out relative completeness since rapid mixing is not expressible in the logic, but it does suggest that the underlying coupling proof technique may be incomplete.

The situation is similar for APRHL. Our privacy proofs often use program transformations to compensate for the incompleteness of the loop rules; these transformations could potentially be avoided given more advanced loop rules or richer reasoning about the privacy parameters ε and δ. However, it is not clear what role the various structural rules (e.g., [PW-EQ]) should play when proving completeness.

Enhancing our proof systems and identifying complete fragments for randomized programs—or even more fundamentally, coming up with sensible notions of completeness for coupling proofs—are intriguing and challenging directions for future theoretical work.

Connecting back to probabilistic bisimulation.   Probabilistic liftings were first developed in the context of probabilistic bisimulation (Larsen and Skou, 1991); it would be interesting to revisit this rich theory in light of our connections. Approximate couplings, which support a multiplicative notion of approximation, appear to be new to the probabilistic bisimulation literature.

New applications

The examples we have seen are drawn from classical coupling proofs in mathematics. While these case studies concisely demonstrate various advanced features of the proof technique, they are perhaps less well-motivated from the perspective of program verification. However, now that formal verification can leverage couplings, we can search for applications to typical verification properties.

At the same time, there remains plenty of room to push the limits of coupling proofs on more theoretical examples, especially using approximate couplings. For example, we only applied approximate couplings for

proving (ε, δ)-differential privacy; variants of approximate couplings for reasoning about f-divergences, like KL-divergence, Hellinger distance, and χ^2 divergence, currently lack concrete applications. Other natural targets include relaxations of differential privacy like *random differential privacy* (RDP) (Hall, Wasserman, and Rinaldo, 2013). For exact couplings, advanced constructions like *coupling from the past* (Propp and Wilson, 1996) and *variable length path couplings* (Hayes and Vigoda, 2007) may suggest interesting ways to enrich relational reasoning. We expect theoretically sophisticated examples will guide the development of formal verification for probabilistic relational properties.

Proof automation

Throughout, we have presented program logic proofs on paper. Such proofs can be formalized in existing prototype implementations of PRHL and APRHL in the EASYCRYPT system (Barthe et al., 2013b), an interactive proof assistant. To make the proof technique more practical, however, more investigation is needed into automating coupling proofs. By eliminating much of the probabilistic reasoning, which pose significant challenges for automated solvers today, coupling proofs may enable automated proofs for programs and properties where even manual, interactive proofs would previously have been quite challenging. Realizing these gains in practice is a natural direction for further investigation.

6.3   Bridging two theories

This thesis represents a confluence of ideas from two theories: coupling proofs from the theory of algorithms, and program logics from the theory of formal verification. While mathematical rigor is a hallmark of both areas, the two fields currently proceed on separate tracks. The theory of algorithms and complexity investigates quantitative aspects of computation, like running time, space usage, and degree of approximation, while the theory of semantics and formal verification explores the compositional structure of programs and how to reason about them. That there should be two distinct theoretical branches is perhaps unsurprising; in many ways, the situation mirrors traditional divisions between analysis and algebra in mathematics. However, what is more surprising is the wide gulf between the two communities today. In many parts of the world, for instance, semantics and verification don't fall under the umbrella term Theoretical Computer Science (TCS).

Our results give a glimpse of the fruitful terrain that lies in between, and the potential gains in applying perspectives and tools from both worlds. Formal verification stands to benefit from understanding how humans reason about algorithms, while algorithms and complexity theory could achieve simpler proofs by generalizing properties and focusing on composition. The time is ripe to bring these theories back into contact, and to see where the conversation leads.

Appendix A

Soundness of $\times$PRHL

We prove soundness of the logic $\times$PRHL presented in Chapter 3, consisting of the logical rules in Figs. 3.1 to 3.3 and the asynchronous loop rule in Fig. 3.4.

We will need a pair of technical lemmas. First, distribution bind commutes with projections.

Lemma A.1.1. *Let $i \in \{1, 2\}$. Given $\mu \in$ **SDistr**$(\mathcal{A}_1 \times \mathcal{A}_2)$ and $f : \mathcal{A}_1 \times \mathcal{A}_2 \to$ **SDistr**$(\mathcal{B}_1 \times \mathcal{B}_2)$, suppose $g_i : \mathcal{A}_i \to$ **SDistr**$(\mathcal{B}_i)$ is such that for all $(a_1, a_2) \in supp(\mu)$, we have $\pi_i(f(a_1, a_2)) \le g_i(a_i)$. Then*

$$\pi_i(\mathbf{bind}(\mu, f)) \le \mathbf{bind}(\pi_i(\mu), g_i).$$

Similarly, if for all $(a_1, a_2) \in supp(\mu)$ we have $\pi_i(f(a_1, a_2)) \ge g_i(a_i)$, then

$$\pi_i(\mathbf{bind}(\mu, f)) \ge \mathbf{bind}(\pi_i(\mu), g_i).$$

Proof. We consider the $\le$ case with $i = 1$; the case $i = 2$ and the $\ge$ cases are similar. Let $\eta \triangleq \pi_1(\mathbf{bind}(\mu, f))$. For any element $h \in \mathcal{B}_1$,

$$
\begin{aligned}
\eta(h) &= \sum_{t \in \mathcal{B}_2} \sum_{(r,s) \in \mathcal{A}_1 \times \mathcal{A}_2} \mu(r,s) \cdot f(r,s)(h,t) \\
&= \sum_{(r,s) \in \mathrm{supp}(\mu)} \mu(r,s) \sum_{t \in \mathcal{B}_2} f(r,s)(h,t) \\
&\le \sum_{(r,s) \in \mathrm{supp}(\mu)} \mu(r,s) \cdot g_1(r)(h) \\
&= \sum_{(r,s) \in \mathcal{A}_1 \times \mathcal{A}_2} \mu(r,s) \cdot g_1(r)(h) \\
&= \sum_{r \in \mathcal{A}_1} \pi_1(\mu)(r) \cdot g_1(r)(h) \\
&= \mathbf{bind}(\pi_1(\mu), g_1)(h). \qquad \square
\end{aligned}
$$

Second, projections commute with monotone limits.

Lemma A.1.2. *Let $\{\mu^{(i)}\}_i$ be a monotonically increasing sequence in **SDistr**$(\mathcal{A}_1 \times \mathcal{A}_2)$ converging to a sub-distribution μ. Then projections commute with limits:*

$$\pi_j \left(\lim_{i \to \infty} \mu^{(i)} \right) = \lim_{i \to \infty} \pi_j \left(\mu^{(i)} \right)$$

for $j \in \{1, 2\}$, and all limits exist.

Proof. By unfolding definitions and applying the monotone convergence theorem, taking the discrete (counting) measure over **State** (see, e.g., Rudin (1976, Theorem 11.28)). ☐

Now we can show soundness of ×PRHL.

Theorem 3.3.1 (Soundness of ×PRHL)**.** *Let ρ be a logical context. If a judgment is derivable*

$$\rho \vdash \left\{ \Phi \right\} \begin{array}{c} c_1 \\ c_2 \end{array} \left\{ \Psi \right\} \blacktriangleright c_\times,$$

then it is valid:

$$\rho \models \left\{ \Phi \right\} \begin{array}{c} c_1 \\ c_2 \end{array} \left\{ \Psi \right\} \blacktriangleright c_\times.$$

Proof. By induction on the height of the proof derivation. In the base case the derivation consists of a single rule with no ×PRHL premises; this rule must be one of the axiom rules: [SKIP], [ASSN], [SAMPLE], or the one-sided variants. In the inductive case, the derivation ends in one of the other rules. By performing a case analysis on the last rule in the derivation, we handle the base and inductive cases together.

We consider the two-sided rules first (Fig. 3.1), followed by the one-sided rules (Fig. 3.2), the structural rules (Fig. 3.3), and finally the asynchronous loop rule (Fig. 3.4). Given soundness for the premises, we show the product program in the conclusion satisfies the support condition and the marginal conditions in Definition 3.1.1. In all cases let m_1, m_2 be two memories that satisfy the pre-condition of the conclusion, let $\mu_\times$ be the output distribution of the product program with input (m_1, m_2), and let $\mu_1 \triangleq \pi_1(\mu_\times)$ and $\mu_2 \triangleq \pi_2(\mu_\times)$ be the two projections of the output distribution. We will leave the logical context ρ implicit when taking the semantics $[\![-]\!]$; the logical variables play no role in the proof.

For the loop rules, recall from Fig. 2.1 that the semantics of a loop **while** e **do** c on initial memory m is defined as the limit of its finite approximants:

$$\mu^{(i)}(m) \triangleq \begin{cases} \bot & : i = 0 \wedge [\![e]\!]m = \textbf{true} \\ \textbf{unit}(m) & : i = 0 \wedge [\![e]\!]m = \textbf{false} \\ \textbf{bind}([\![\textbf{if } e \textbf{ then } c]\!]m, \mu^{(i-1)}) & : i > 0. \end{cases}$$

Case [SKIP] Trivial.

Case [ASSN] The support condition is clear since all program variables in $e_1\langle 1 \rangle, e_2\langle 2 \rangle$ are tagged with $\langle 1 \rangle, \langle 2 \rangle$ respectively. The marginal conditions are clear as well: given any two input memories satisfying the pre-condition, the two output memories from c_1 and c_2 are point distributions where x_1 is updated to e_1 and x_2 is updated to e_2.

Case [SAMPLE] The support of $\mu_\times$ lies in

$$\{(m_1', m_2') \mid \exists v, \; m_1'(x_1) = v \wedge m_2'(x_2) = f(v)\}.$$

Since all output memories (m_1', m_2') in the support are equal to the input memories (m_1, m_2) on all variables besides x_1 and x_2, the support condition is clear.

Now recall that all primitive distributions d_1, d_2 are uniform over finite sets. Hence $\text{supp}(d_1)$ and $\text{supp}(d_2)$ are finite, and since there is a bijection $f : \text{supp}(d_1) \rightarrow \text{supp}(d_2)$, the supports have the same size n. For every $v \in \text{supp}(d_1)$, we have

$$\mu_1(m_1[x_1 \mapsto v]) = 1/n$$

and $\mu_1(m') = 0$ otherwise. By the semantics of the sampling command, $\mu_1 = [\![c_1]\!]m_1$ so the first marginal condition is satisfied. Since f is injective, for every $v \in \mathrm{supp}(d_1)$ we have

$$\mu_2(m_2[x_2 \mapsto f(v)]) = 1/n.$$

and $\mu_2(m') = 0$ otherwise. Since f is surjective, for every $v \in \mathrm{supp}(d_2)$ we have

$$\mu_2(m_2[x_2 \mapsto v]) = 1/n,$$

giving $\mu_2 = [\![c_2]\!]m_2$ and the second marginal condition.

Case [SEQ] Let the product programs in the premises be $c_\times$ and $c'_\times$. By induction, these product programs satisfy the support and marginal conditions for their respective judgments. To establish the conclusion, the support condition is clear: by induction, the support of $[\![c_\times]\!](m_1, m_2)$ lies in $[\![\Psi]\!]$ and for any $(m'_1, m_2)' \in [\![\Psi]\!]$, the support of $[\![c'_\times]\!](m'_1, m'_2)$ lies in $[\![\Theta]\!]$.

It remains to show the marginal conditions. For $i \in \{1, 2\}$,

$$
\begin{aligned}
\mu_i &= \pi_i([\![c_\times; c'_\times]\!](m_1, m_2)) \\
&= \pi_i(\mathbf{bind}([\![c_\times]\!](m_1, m_2), [\![c'_\times]\!])) && \text{(semantics)} \\
&= \mathbf{bind}(\pi_i([\![c_\times]\!](m_1, m_2)), [\![c'_i]\!]) && \text{(Lemma A.1.1 and induction)} \\
&= \mathbf{bind}([\![c_i]\!]m_i, [\![c'_i]\!]) && \text{(induction)} \\
&= [\![c_i; c'_i]\!]m_i. && \text{(semantics)}
\end{aligned}
$$

Case [COND] Let $c_\times, c'_\times$ be the two product programs for the two premises. There are two cases: either e_1 is true in the first initial memory m_1, or not. (Since (m_1, m_2) satisfy the pre-condition Φ, these two cases correspond to e_2 being true and false in the second initial memory m_2.)

Suppose e_1 is true in m_1. Then $e_1\langle 1\rangle$ is true in (m_1, m_2) and the product program is equivalent to simply executing $c_\times$ on (m_1, m_2). Since the two initial memories (m_1, m_2) satisfy Φ, by induction on the first premise, the support of the product program lies in $[\![\Psi]\!]$ and the marginals satisfy

$$\mu_1 = \pi_1([\![c_1]\!]m_1) \quad \text{and} \quad \mu_2 = \pi_2([\![c_2]\!]m_2).$$

Since $e_1\langle 1\rangle$ and $e_2\langle 2\rangle$ are both true in (m_1, m_2), we also have

$$\mu_1 = \pi_1([\![\mathbf{if}\ e_1\ \mathbf{then}\ c_1\ \mathbf{else}\ c'_1]\!]m_1) \quad \text{and} \quad \mu_2 = \pi_2([\![\mathbf{if}\ e_2\ \mathbf{then}\ c_2\ \mathbf{else}\ c'_2]\!]m_2).$$

Hence, both the marginal and support conditions hold when e_1 is true in m_1.

The other case, where $e_1\langle 1\rangle$ and $e_2\langle 2\rangle$ are false in (m_1, m_2), follows by the second premise.

Case [WHILE] Let the product program in the conclusion be $\mathbf{while}\ e_1\langle 1\rangle\ \mathbf{do}\ c_\times$ and let $\mu^{(i)}(m_1, m_2)$ be its i-th approximants. Define $\mu_1^{(i)} \triangleq \pi_1 \circ \mu^{(i)}, \mu_2^{(i)} \triangleq \pi_2 \circ \mu^{(i)}$ to be the first and second marginals of the approximants, and $\eta_1^{(i)}, \eta_2^{(i)}$ to be the i-th approximants of the loops $\mathbf{while}\ e_1\ \mathbf{do}\ c_1$ and $\mathbf{while}\ e_2\ \mathbf{do}\ c_2$, respectively.

Let's consider the support condition first. We prove if (m_1, m_2) satisfies Φ, then $\mu^{(i)}(m_1, m_2)$ has support contained in $[\![\Phi \wedge \neg e_1\langle 1\rangle]\!]$ for every i by induction. The base case $i = 0$ is clear. For the inductive step $i > 0$ there are two cases. If $e_1\langle 1\rangle$ is false in (m_1, m_2), then $\mu^{(i)}(m_1, m_2) = \mathbf{unit}(m_1, m_2)$. Otherwise if $e_1\langle 1\rangle$ is true, then

$$\mu^{(i)}(m_1, m_2) = \mathbf{bind}([\![c_\times]\!](m_1, m_2), \mu^{(i-1)}).$$

By the outer induction hypothesis applied to the premise, the support of $[\![c_\times]\!](m_1, m_2)$ lies in $[\![\Phi \wedge \neg e_1\langle 1 \rangle]\!]$. The inner induction hypothesis applied to $\mu^{(i-1)}$ shows $\mu^{(i)}(m_1, m_2)$ also has support in $[\![\Phi \wedge \neg e_1\langle 1 \rangle]\!]$, completing the inner induction. Since this holds for all i, the limit sub-distribution

$$\lim_{i \to \infty} \mu^{(i)}(m_1, m_2) = [\![\textbf{while } e_1\langle 1 \rangle \textbf{ do } c_\times]\!](m_1, m_2)$$

also has support in $[\![\Phi \wedge \neg e_1\langle 1 \rangle]\!]$ as desired.

Next, we turn to the marginal conditions. We first show the projections of the approximants of the product program are equal to the approximants for the individual programs, concluding the marginal conditions in the limit. Let (m_1, m_2) be memories satisfying Φ. We claim $\pi_1(\mu^{(i)}(m_1, m_2)) = \eta_1^{(i)}(m_1)$ and $\pi_2(\mu^{(i)}(m_1, m_2)) = \eta_2^{(i)}(m_2)$ for every i.

The claim follows by induction on i. The base case $i = 0$ is immediate—since (m_1, m_2) satisfy Φ, either $e_1\langle 1 \rangle = e_2\langle 2 \rangle = \textbf{true}$ or $e_1\langle 1 \rangle = e_2\langle 2 \rangle = \textbf{false}$. The inductive step $i > 0$ is more interesting. Unrolling the approximants one step, we have

$$\mu^{(i)}(m_1, m_2) = \textbf{bind}([\![\textbf{if } e_1\langle 1 \rangle \textbf{ then } c_\times]\!](m_1, m_2), \mu^{(i-1)})$$
$$\eta_1^{(i)}(m_1) = \textbf{bind}([\![\textbf{if } e_1 \textbf{ then } c_1]\!]m_1, \eta_1^{(i-1)})$$
$$\eta_2^{(i)}(m_2) = \textbf{bind}([\![\textbf{if } e_2 \textbf{ then } c_2]\!]m_2, \eta_2^{(i-1)}).$$

If $e_1\langle 1 \rangle$ is false in (m_1, m_2), then all three conditionals are equivalent to **skip** so

$$\mu^{(i)} = \mu^{(i-1)} \quad \text{and} \quad \eta_1^{(i)} = \eta_1^{(i-1)} \quad \text{and} \quad \eta_2^{(i)} = \eta_2^{(i-1)};$$

we conclude by the inductive hypothesis. Otherwise, $e_1\langle 1 \rangle$ and $e_2\langle 2 \rangle$ are true in (m_1, m_2) so the same branch is taken in all three approximants:

$$\mu^{(i)}(m_1, m_2) = \textbf{bind}([\![c_\times]\!](m_1, m_2), \mu^{(i-1)})$$
$$\eta_1^{(i)}(m_1) = \textbf{bind}([\![c_1]\!]m_1, \eta_1^{(i-1)})$$
$$\eta_2^{(i)}(m_2) = \textbf{bind}([\![c_2]\!]m_2, \eta_2^{(i-1)}).$$

By the outer induction hypothesis on the premise of the rule (noting that e_1 is true in m_1),

$$\pi_1([\![c_\times]\!](m_1, m_2)) = [\![c_1]\!]m_1 \quad \text{and} \quad \pi_2([\![c_\times]\!](m_1, m_2)) = [\![c_2]\!]m_2.$$

By the inner induction hypothesis, $\mu^{(i-1)}(m_1, m_2)$ has projections $\eta_1^{(i-1)}(m_1)$ and $\eta_2^{(i-1)}(m_2)$, so Lemma A.1.1 gives

$$\pi_1(\mu^{(i)}(m_1, m_2)) = \eta_1^{(i)}(m_1) \quad \text{and} \quad \pi_2(\mu^{(i)}(m_1, m_2)) = \eta_2^{(i)}(m_2)$$

for every i. Taking the limit as i tends to ∞, we have the marginal conditions

$$[\![\textbf{while } e_j \textbf{ do } c_j]\!](m_j) \triangleq \lim_{i \to \infty} \eta_j^{(i)}(m_j)$$
$$= \lim_{i \to \infty} \pi_j(\mu^{(i)}(m_j))$$
$$= \pi_j\left(\lim_{i \to \infty} \mu^{(i)}(m_j)\right)$$
$$\triangleq \pi_j([\![\textbf{while } e_1\langle 1 \rangle \textbf{ do } c_\times]\!](m_1, m_2))$$

for $j = \{1, 2\}$. (We may interchange marginals and limits by Lemma A.1.2 since $\{\mu^{(i)}(m_j)\}_i$ is monotonically increasing by definition.)

Case [ASSN-L] ([ASSN-R] similar) Trivial.

Case [SAMPLE-L] ([SAMPLE-R] similar) Let d_1 have support with size n. The support condition is clear. For the marginal condition, note

$$\mu_\times(m_1[x_1 \mapsto v], m_2) = 1/n$$

for every $v \in \text{supp}(d_1)$, and zero otherwise. Hence,

$$\mu_1(m_1([x_1 \mapsto v]) = 1/n$$

for every $v \in \text{supp}(d_1)$, and zero otherwise, while μ_2 is the point distribution at m_2. The semantics of $x_1 \xleftarrow{\$} d_1$ and **skip** gives the marginal conditions.

Case [COND-L] ([COND-R] similar) There are two cases: either $e_1\langle 1 \rangle$ is true in (m_1, m_2), or not. On input (m_1, m_2), the product program has the same semantics as c and c' in the respective cases, hence the support condition follows by induction using the support condition in the first and second premises respectively.

The marginal conditions are similar. If $e_1\langle 1 \rangle$ is true in (m_1, m_2), then the product program has the same semantics as c, and the first program **if** e_1 **then** c_1 **else** c_1' has the same semantics as c_1. Hence, the marginal conditions follow by induction using the marginal condition from the first premise.

In the other case, $e_1\langle 1 \rangle$ is false in (m_1, m_2) and the product program has the same semantics as c', and on m_1 the first program **if** e_1 **then** c_1 **else** c_1' has the same semantics as c_1'. Hence, the marginal conditions follow by induction using the marginal condition from the second premise.

Case [WHILE-L] ([WHILE-R] similar) Let the final product program be **while** $e_1\langle 1 \rangle$ **do** $c_\times$ with i-th approximants $\mu^{(i)}(m_1, m_2)$. Define $\mu_1^{(i)} \triangleq \pi_1 \circ \mu^{(i)}, \mu_2^{(i)} \triangleq \pi_2 \circ \mu^{(i)}$ to be the first and second marginals of the approximants, and $\eta^{(i)}$ to be the i-th approximants of the loop **while** e_1 **do** c_1.

For the support condition, we show if (m_1, m_2) satisfies Φ then $\mu^{(i)}(m_1, m_2)$ has support contained in $[\![\Phi \wedge \neg e_1\langle 1 \rangle]\!]$ for every i by induction on i. The base case $i = 0$ is clear. For the inductive step $i > 0$, there are two cases. If $e_1\langle 1 \rangle$ is false in (m_1, m_2), then $\mu^{(i)}(m_1, m_2) = \textbf{unit}(m_1, m_2)$ and we are done. Otherwise if $e_1\langle 1 \rangle$ is true, then

$$\mu^{(i)}(m_1, m_2) = \textbf{bind}([\![c_\times]\!](m_1, m_2), \mu^{(i-1)}).$$

By the outer induction hypothesis applied to the premise, the support of $[\![c_\times]\!](m_1, m_2)$ lies in $[\![\Phi \wedge \neg e_1\langle 1 \rangle]\!]$. The inner induction hypothesis applied to $\mu^{(i-1)}$ implies $\mu^{(i)}(m_1, m_2)$ also has support contained in $[\![\Phi \wedge \neg e_1\langle 1 \rangle]\!]$, completing the inner induction. Since this is true for all i, the limit sub-distribution

$$\lim_{i \to \infty} \mu^{(i)}(m_1, m_2) = [\![\textbf{while } e_1\langle 1 \rangle \textbf{ do } c_\times]\!](m_1, m_2)$$

also has support in $[\![\Phi \wedge \neg e_1\langle 1 \rangle]\!]$ as desired.

Now we turn to the marginal conditions. We show the projections of the approximant of the product program are equal to the approximants for the individual programs, concluding the marginal conditions in the limit. Let (m_1, m_2) be any memories satisfying Φ. We claim $\pi_1(\mu^{(i)}(m_1, m_2)) = \eta_1^{(i)}(m_1)$ and $\pi_2(\mu^{(i)}(m_1, m_2))$ is a point sub-distribution with all mass on m_2, for every i.

The claim follows by induction on i. The base case $i = 0$ is clear—$e_1\langle 1 \rangle$ is either true or false. If $e_1\langle 1 \rangle$ is true, $\mu^{(i)} = \textbf{unit}(m_1, m_2)$, $\eta_1^{(i)} = \textbf{unit}(m_1)$, and $\pi_2(\mu^{(i)}) = \textbf{unit}(m_2)$. If $e_1\langle 1 \rangle$ is false, then all approximants are the zero sub-distribution $\perp$.

The inductive step $i > 0$ is more interesting. Unrolling the approximants one step, we have

$$\mu^{(i)}(m_1, m_2) = \textbf{bind}([\![\textbf{if } e_1\langle 1 \rangle \textbf{ then } c_\times]\!](m_1, m_2), \mu^{(i-1)})$$

$$\eta_1^{(i)}(m_1) = \mathbf{bind}(\llbracket \mathbf{if}\ e_1\ \mathbf{then}\ c_1 \rrbracket m_1, \eta_1^{(i-1)}).$$

If $e_1\langle 1\rangle$ is false in (m_1, m_2), then both conditionals are equivalent to **skip**. Hence

$$\mu^{(i)} = \mu^{(i-1)} \quad \text{and} \quad \eta_1^{(i)} = \eta_1^{(i-1)},$$

and we conclude by the induction on i. Otherwise, $e_1\langle 1\rangle$ is true in (m_1, m_2). In this case, the conditional branch is taken in both programs, so

$$\mu^{(i)}(m_1, m_2) = \mathbf{bind}(\llbracket c_\times \rrbracket(m_1, m_2), \mu^{(i-1)})$$
$$\eta_1^{(i)}(m_1) = \mathbf{bind}(\llbracket c_1 \rrbracket m_1, \eta_1^{(i-1)}).$$

By the outer induction hypothesis on the premise of the rule (noting that $e_1\langle 1\rangle$ is true),

$$\pi_1(\llbracket c_\times \rrbracket(m_1, m_2)) = \llbracket c_1 \rrbracket m_1$$
$$\pi_2(\llbracket c_\times \rrbracket(m_1, m_2)) = \llbracket \mathbf{skip} \rrbracket m_2 = \mathbf{unit}(m_2).$$

The inner induction hypothesis shows the first marginal of $\mu^{(i-1)}(m_1, m_2)$ is $\eta_1^{(i-1)}(m_1)$; Lemma A.1.1 establishes $\pi_1(\mu^{(i)}(m_1, m_2)) = \eta_1^{(i)}(m_1)$. Similarly, by the inner induction hypothesis showing the second marginal of $\mu^{(i-1)}(m_1, m_2)$ is a point mass at m_2, we establish the same for the second marginal of $\mu^{(i)}(m_1, m_2)$. Furthermore, since the weight of a sub-distribution is preserved under projections, we also know $\pi_2(\mu^{(i)}(m_1, m_2))$ is a point sub-distribution at m_2 with weight $|\mu^{(i)}(m_1, m_2)|$.

Now we take limits to obtain the first marginal condition:

$$\eta_1(m_1) = \lim_{i\to\infty} \eta_1^{(i)}(m_j) = \lim_{i\to\infty} \pi_1(\mu^{(i)}(m_1, m_2)) = \pi_1\left(\lim_{i\to\infty} \mu^{(i)}(m_1, m_2)\right) = \pi_1(\mu_\times),$$

interchanging limits and projections by Lemma A.1.2, since $\{\mu^{(i)}(m_1, m_2)\}_i$ is monotonically increasing.

For the second marginal we have

$$\pi_2(\mu_\times) = \mathbf{unit}(m_2) \cdot |\mu_\times|.$$

By the premise, the loop **while** e_1 **do** c_1 is lossless. Hence,

$$1 = |\eta_1(m_1)| = |\mu_\times|$$

and the second projection of $\mu_\times$ is simply $\mathbf{unit}(m_2) = \llbracket \mathbf{skip} \rrbracket m_2$ as claimed.

Case [CONSEQ] Trivial.

Case [EQUIV] Trivial.

Case [CASE] By case analysis on whether e is true in (m_1, m_2), using essentially the same reasoning as in [CASE], [COND-L], or [COND-R].

Case [FRAME] The marginal conditions are clear by induction. Let V be the set of variables that are not in $\mathrm{MV}(c)$. Since Θ has free variables in V, we can interpret Θ as a predicate on memories restricted to V. Then initially $(m_1[V], m_2[V]) \in \llbracket \Theta \rrbracket$. Since c does not modify variables in V, the support of $\mu_\times$ is contained in

$$\{(m_1', m_2') \mid m_1'[V] = m_1[V] \wedge m_2'[V] = m_2[V]\} \subseteq \llbracket \Theta \rrbracket.$$

Hence the support condition is satisfied as well.

Case [WHILE-GEN] We label the premises for easy reference:

$$\models \Phi \to (e_1 \langle 1 \rangle \vee e_2 \langle 2 \rangle) = e \tag{A.1}$$

$$\models \Phi \wedge e \to p_0 \oplus p_1 \oplus p_2 \tag{A.2}$$

$$\models \Phi \wedge p_0 \wedge e \to e_1 \langle 1 \rangle = e_2 \langle 2 \rangle \tag{A.3}$$

$$\models \Phi \wedge p_1 \wedge e \to e_1 \langle 1 \rangle \wedge \Phi_1 \langle 1 \rangle \tag{A.4}$$

$$\models \Phi \wedge p_2 \wedge e \to e_2 \langle 2 \rangle \wedge \Phi_2 \langle 2 \rangle \tag{A.5}$$

$$\Phi_1 \models \textbf{while } e_1 \wedge p_1 \textbf{ do } c_1 \text{ lossless} \tag{A.6}$$

$$\Phi_2 \models \textbf{while } e_2 \wedge p_2 \textbf{ do } c_2 \text{ lossless} \tag{A.7}$$

$$\vdash \left\{ \Phi \wedge e \wedge p_0 \right\} \begin{array}{c} (\textbf{if } e_1 \textbf{ then } c_1)^{K_1} \\ (\textbf{if } e_2 \textbf{ then } c_2)^{K_2} \end{array} \left\{ \Phi \right\} \blacktriangleright c_0' \tag{A.8}$$

$$\vdash \left\{ \Phi \wedge e_1 \wedge p_1 \right\} \begin{array}{c} c_1 \\ \textbf{skip} \end{array} \left\{ \Phi \right\} \blacktriangleright c_1' \tag{A.9}$$

$$\vdash \left\{ \Phi \wedge e_2 \wedge p_2 \right\} \begin{array}{c} \textbf{skip} \\ c_2 \end{array} \left\{ \Phi \right\} \blacktriangleright c_2' \tag{A.10}$$

Let $\theta_\times$ be the semantics of the product program in the conclusion and let $\theta^{(i)}$ be its i-th approximants. For the support condition, we first show

$$\text{supp}(\theta^{(i)}(a_1, a_2)) \subseteq [\![\Phi \wedge \neg e_1 \langle 1 \rangle \wedge \neg e_2 \langle 2 \rangle]\!]$$

for every i and (a_1, a_2) satisfying Φ. The proof is by induction on i. The base case $i = 0$ is clear: if e is false in (a_1, a_2) then $\theta^{(0)}(a_1, a_2) = \textbf{unit}(a_1, a_2)$, otherwise if e is true then $\theta^{(0)}(a_1, a_2) = \bot$, so in both cases we have the desired support.

For the inductive step $i > 0$, if e is false in (a_1, a_2) then $\theta^{(i)}(a_1, a_2) = \textbf{unit}(a_1, a_2)$ and the support condition is satisfied. Otherwise, we unfold the product program one step giving three cases:

$$\theta^{(i)}(a_1, a_2) = \begin{cases} \textbf{bind}([\![c_0']\!](a_1, a_2), \theta^{(i-1)}) & : [\![p_0]\!](a_1, a_2) = \textbf{true} \\ \textbf{bind}([\![c_1']\!](a_1, a_2), \theta^{(i-1)}) & : [\![p_1]\!](a_1, a_2) = \textbf{true} \\ \textbf{bind}([\![c_2']\!](a_1, a_2), \theta^{(i-1)}) & : [\![p_2]\!](a_1, a_2) = \textbf{true}. \end{cases}$$

Exactly one of the three cases holds, by Eq. (A.2). By the outer induction hypothesis, the premises of the rule (Eqs. (A.8) to (A.10)) show that in the three cases, the corresponding product program c_0', c_1', c_2' on input memory (a_1, a_2) produces a sub-distribution with support in $[\![\Phi]\!]$. Hence $\theta^{(i)}(a_1, a_2)$ has the desired support using the inner induction hypothesis on $\theta^{(i-1)}$. Passing to the limit, we conclude the support condition:

$$\text{supp}(\theta_\times(m_1, m_2)) = \text{supp}\left(\lim_{i \to \infty} \theta^{(i)}(m_1, m_2) \right) \subseteq [\![\Phi \wedge \neg e_1 \langle 1 \rangle \wedge \neg e_2 \langle 2 \rangle]\!].$$

Next, we turn to the marginal conditions. Let $\eta_1, \eta_2 : \textbf{State} \to \textbf{SDistr}(\textbf{State})$ be the semantics of the loops $\textbf{while } e_1 \textbf{ do } c_1$ and $\textbf{while } e_2 \textbf{ do } c_2$, and let $\eta_1^{(i)}, \eta_2^{(i)} : \textbf{State} \to \textbf{SDistr}(\textbf{State})$ be their respective i-th approximants. We show for every i and every (a_1, a_2) satisfying the invariant Φ, we have

$$\eta_1^{(i)}(a_1) \leq \pi_1(\theta_\times(a_1, a_2)) \tag{A.11}$$

$$\pi_1(\theta^{(i)}(a_1, a_2)) \leq \eta_1(a_1) \tag{A.12}$$

$$\eta_2^{(i)}(a_2) \leq \pi_2(\theta_\times(a_1, a_2)) \tag{A.13}$$

$$\pi_2(\theta^{(i)}(a_1, a_2)) \le \eta_2(a_2). \tag{A.14}$$

Taking limits as i tends to infinity will give the desired marginal conditions.

We begin with Eq. (A.11) by induction on i. For the base case $i = 0$, if e is false in (a_1, a_2) then both sides are equal to $\mathbf{unit}(a_1)$. Otherwise, if e and $e_1\langle 1 \rangle$ are true, then both sides are equal to $\bot$. Finally, if e is true and $e_1\langle 1 \rangle$ is false, then $\eta_1^{(0)}(a_1) = \mathbf{unit}(a_1)$ by Eq. (A.1). In this case, $e_2\langle 2 \rangle$ must be true. By Eq. (A.5), we are in case p_2 and the product program executes c_2'. By the marginal condition from premise Eq. (A.10), c_2' preserves a_1 so $e_1\langle 1 \rangle$ remains false. Hence,

$$\theta_\times(a_1, a_2) = [\![\mathbf{while}\ e_2\langle 2 \rangle \wedge p_2\ \mathbf{do}\ c_2']\!](a_1, a_2).$$

By reasoning analogous to the case [WHILE-R] with Eq. (A.7) and the outer inductive hypothesis on Eq. (A.10), we have the marginal condition

$$\pi_1(\theta_\times(a_1, a_2)) = \mathbf{unit}(a_1) = \eta_1^{(0)}(a_1)$$

establishing the base case.

Next, we consider the inductive case $i > 0$. If e is false in (a_1, a_2), then $e_1\langle 1 \rangle$ is false in a_1 and hence $\eta_1^{(i)}(a_1) = \pi_1(\theta_\times(a_1, a_2)) = \mathbf{unit}(a_1)$. Otherwise, e is true and there are three subcases.

Subcase for Eq. (A.11): p_0 is true. If p_0 is true in (a_1, a_2), then $e_1\langle 1 \rangle = e_2\langle 2 \rangle$ are also true by Eq. (A.3). First, suppose $i = K_1$. Unrolling the loops gives

$$\theta_\times(a_1, a_2) = \mathbf{bind}([\![c_0']\!](a_1, a_2), \theta_\times)$$
$$\eta_1^{(K_1)}(a_1) = \mathbf{bind}([\![(\mathbf{if}\ e_1\ \mathbf{then}\ c_1)^{K_1}]\!]a_1, \eta_1^{(0)}).$$

The marginal condition from the induction hypothesis on premise Eq. (A.8) gives

$$\pi_1([\![c_0']\!](a_1, a_2)) = [\![(\mathbf{if}\ e_1\ \mathbf{then}\ c_1)^{K_1}]\!](a_1);$$

by the support condition, $\mathrm{supp}([\![c_0']\!](a_1, a_2)) \subseteq [\![\Phi]\!]$. Furthermore, the base case for the inner induction yields $\eta_1^{(0)}(b_1) \le \pi_1(\theta_\times(b_1, b_2))$ for every $(b_1, b_2) \in [\![\Phi]\!]$, so Lemma A.1.1 gives

$$\eta^{(K_1)}(a_1) \le \pi_1(\theta_\times(a_1, a_2)).$$

Now suppose $i < K_1$. From the previous case and monotonicity, we have

$$\eta_1^{(i)}(a_1) \le \eta_1^{(K_1)}(a_1) \le \pi_1(\theta_\times(a_1, a_2)).$$

With the cases $i \le K_1$ covered, we turn to the remaining cases $i > K_1$. Unrolling the loops:

$$\theta_\times(a_1, a_2) = \mathbf{bind}([\![c_0']\!](a_1, a_2), \theta_\times)$$
$$\eta_1^{(i)}(a_1) = \mathbf{bind}([\![(\mathbf{if}\ e_1\ \mathbf{then}\ c_1)^{K_1}]\!]a_1, \eta_1^{(i-K_1)}).$$

The marginal condition from the induction hypothesis on premise Eq. (A.8) gives

$$\pi_1([\![c_0']\!](a_1, a_2)) = [\![(\mathbf{if}\ e_1\ \mathbf{then}\ c_1)^{K_1}]\!]a_1;$$

by the support condition, we have $[\![c_0']\!](a_1, a_2) \subseteq [\![\Phi]\!]$. Furthermore, by the inner inductive hypothesis for $\eta^{(i-K_1)}$ we have $\eta_1^{(i-K_1)}(b_1) \le \pi_1(\theta_\times(b_1, b_2))$ for every $(b_1, b_2) \in [\![\Phi]\!]$, so Lemma A.1.1 shows

$$\eta^{(i)}(a_1) \le \pi_1(\theta_\times(a_1, a_2))$$

as desired.

Subcase for Eq. (A.11): p_1 **is true.** If p_1 is true in (a_1, a_2), then $e_1\langle 1\rangle$ is also true by Eq. (A.4). Unrolling the loops:

$$\theta_\times(a_1, a_2) = \mathbf{bind}(\llbracket c_1' \rrbracket(a_1, a_2), \theta_\times)$$
$$\eta_1^{(i)}(a_1) = \mathbf{bind}(\llbracket c_1 \rrbracket a_1, \eta_1^{(i-1)}).$$

The induction hypothesis on premise Eq. (A.9) gives $\pi_1(\llbracket c_1' \rrbracket(a_1, a_2)) = \llbracket c_1 \rrbracket a_1$; by the support condition, we also have $\llbracket c_1' \rrbracket(a_1, a_2) \subseteq \llbracket \Phi \rrbracket$. Furthermore, by the inner induction hypothesis we have $\eta_1^{(i-1)}(b_1) \leq \pi_1(\theta_\times(b_1, b_2))$ for every $(b_1, b_2) \in \llbracket \Phi \rrbracket$, so Lemma A.1.1 yields

$$\eta^{(i)}(a_1) \leq \pi_1(\theta_\times(a_1, a_2)).$$

Subcase for Eq. (A.11): p_2 **is true.** If p_2 is true in (a_1, a_2), then $e_2\langle 2\rangle$ is true by Eq. (A.5). Define

$$\eta \triangleq \llbracket \mathbf{while}\ e \wedge p_2\ \mathbf{do}\ c_2' \rrbracket(a_1, a_2).$$

We show the equivalence

$$\llbracket \mathbf{while}\ e \wedge p_2\ \mathbf{do}\ c_2' \rrbracket(a_1, a_2) = \llbracket \mathbf{while}\ e_2\langle 2\rangle \wedge p_2\ \mathbf{do}\ c_2' \rrbracket(a_1, a_2) \tag{A.15}$$

by taking the approximants $\sigma^{(i)}$ and $\tau^{(i)}$ of the left and right sides and proving

$$\sigma^{(i)}(b_1, b_2) = \tau^{(i)}(b_1, b_2)$$

for every $(b_1, b_2) \in \llbracket \Phi \rrbracket$. The proof is by induction on i, using $\mathrm{supp}(\tau^{(i)})(b_1, b_2) \subseteq \llbracket \Phi \rrbracket$ from the support condition from premise Eq. (A.10), and

$$\models \Phi \rightarrow (e \wedge p_2 \leftrightarrow e_2\langle 2\rangle \wedge p_2)$$

from Eqs. (A.1) and (A.5).

Using the equivalence Eq. (A.15), we can transform η and show the following:

$$\mathrm{supp}(\eta) \subseteq \llbracket \Phi \wedge \neg(e \wedge p_2) \rrbracket \tag{A.16}$$
$$\pi_1(\eta) = \mathbf{unit}(a_1) \tag{A.17}$$

Both points follow by reasoning similar to the case for [WHILE-R], using premise Eq. (A.10) and the lossless condition Eq. (A.7). The first point also uses $\mathrm{supp}(\eta) \subseteq \llbracket \neg(e \wedge p_2) \rrbracket$, by definition of η.

Returning to the sub-case, if $e_1\langle 1\rangle$ is true, unrolling the product program gives

$$\theta_\times(a_1, a_2) = \mathbf{bind}(\llbracket \mathbf{while}\ e \wedge p_2\ \mathbf{do}\ c_2' \rrbracket(a_1, a_2), \theta_\times).$$

Since the guard $e \wedge p_2$ is false in $\mathrm{supp}(\eta)$ and e_1 is true in the first initial memory a_1, Eq. (A.16) gives

$$\mathrm{supp}(\eta) \subseteq \llbracket \Phi \wedge e_1\langle 1\rangle \wedge \neg(e \wedge p_2) \rrbracket \subseteq \llbracket \Phi \wedge e \wedge \neg p_2 \rrbracket$$

where the second inclusion is because e_1 implies e (by Eq. (A.1)), so p_2 must be false. By Eq. (A.2) either p_0 or p_1 must be true in the support of η. Using Eq. (A.17) to show $\eta(r_1, r_2) > 0$ only when $r_1 = a_1$, we compute:

$$\pi_1(\theta_\times(a_1, a_2)) = \pi_1 \left(\sum_{(r_1, r_2) \in \llbracket \Phi \wedge e \wedge \neg p_2 \rrbracket} \eta(r_1, r_2) \cdot \theta_\times(r_1, r_2) \right)$$

$$= \pi_1 \left(\sum_{r_2:(a_1,r_2)\in[\![\Phi \wedge e \wedge \neg p_2]\!]} \eta(a_1,r_2) \cdot \theta_\times(a_1,r_2) \right)$$

$$= \sum_{r_2:(a_1,r_2)\in[\![\Phi \wedge e \wedge \neg p_2]\!]} \eta(a_1,r_2) \cdot \pi_1(\theta_\times(a_1,r_2))$$

$$= \sum_{r_2:(a_1,r_2)\in[\![\Phi \wedge e \wedge p_0]\!]} \eta(a_1,r_2) \cdot \pi_1(\theta_\times(a_1,r_2))$$

$$+ \sum_{r_2:(a_1,r_2)\in[\![\Phi \wedge e \wedge p_1]\!]} \eta(a_1,r_2) \cdot \pi_1(\theta_\times(a_1,r_2))$$

$$\geq \sum_{r_2:(a_1,r_2)\in[\![\Phi \wedge e \wedge p_0]\!]} \eta(a_1,r_2) \cdot \eta_1^{(i)}(a_1) + \sum_{r_2:(a_1,r_2)\in[\![\Phi \wedge e \wedge p_1]\!]} \eta(a_1,r_2) \cdot \eta_1^{(i)}(a_1)$$

$$= \left(\sum_{r_2:(a_1,r_2)\in[\![\Phi]\!]} \eta(a_1,r_2) \right) \cdot \eta_1^{(i)}(a_1)$$

where on the third line we interchange projection and the sum by Lemma A.1.2, and the inequality is from the cases where p_0 and p_1 are true. Equations (A.16) and (A.17) show

$$\sum_{r_2:(a_1,r_2)\in[\![\Phi]\!]} \eta(a_1,r_2) = |\pi_1(\eta)| = 1$$

and so $\pi_1(\theta_\times(a_1,a_2)) \geq \eta_1^{(i)}(a_1)$ as desired.

If $e_1\langle 1 \rangle$ is false, unrolling the loops gives

$$\theta_\times(a_1,a_2) = \mathbf{bind}([\![\mathbf{while}\ e \wedge p_2\ \mathbf{do}\ c_2']\!](a_1,a_2), \theta_\times)$$
$$\eta_1^{(i)}(a_1) = \mathbf{bind}([\![\mathbf{skip}]\!]a_1, \eta_1^{(i-1)}).$$

Equation (A.17) implies

$$\pi_1([\![\mathbf{while}\ e \wedge p_2\ \mathbf{do}\ c_2']\!](a_1,a_2)) = \pi_1(\eta) = \mathbf{unit}(a_1) = [\![\mathbf{skip}]\!]a_1.$$

Furthermore Eq. (A.16) implies $\mathrm{supp}(\eta) \subseteq [\![\Phi]\!]$, so we apply Lemma A.1.1 with the induction hypothesis $\eta_1^{(i-1)}(b_1) \leq \theta_\times(b_1,b_2)$ for all $(b_1,b_2) \in [\![\Phi]\!]$ to conclude

$$\eta_1^{(i)}(a_1) \leq \pi_1(\theta_\times(a_1,a_2)).$$

This completes the inductive case $i > 0$, establishing Eq. (A.11).

Next, we establish Eq. (A.12) by induction on i. For the base case $i = 0$, if e is true in (a_1,a_2) then

$$\theta^{(0)}(a_1,a_2) = \bot \leq \eta_1(a_1).$$

Otherwise if e is false, then e_1 must be false in a_1 as well and so

$$\pi_1(\theta^{(0)}(a_1,a_2)) = a_1 = \eta_1(a_1).$$

Now we consider the inductive step $i > 0$. Again if e is false in (a_1,a_2) then both sides are $\bot$ and the claim is clear. Otherwise if e is true, there are three cases.

Subcase for Eq. (A.12): p_0 is true. If p_0 is true, then we unfold the loops:

$$\theta^{(i)}(a_1, a_2) = \mathbf{bind}(\llbracket c'_0 \rrbracket(a_1, a_2), \theta^{(i-1)})$$
$$\eta_1(a_1) = \mathbf{bind}(\llbracket (\mathbf{if}\ e_1\ \mathbf{then}\ c_1)^{K_1} \rrbracket(a_1), \eta_1).$$

By the induction hypothesis, for every $(b_1, b_2) \in \llbracket \Phi \rrbracket$ we have $\pi_1(\theta^{(i-1)}(b_1, b_2)) \leq \eta_1(b_1)$. By the marginal condition from the outer induction hypothesis for the premise Eq. (A.8), we also have

$$\pi_1(\llbracket c'_0 \rrbracket(a_1, a_2)) = \llbracket (\mathbf{if}\ e_1\ \mathbf{then}\ c_1)^{K_1} \rrbracket a_1.$$

The support condition from the same induction hypothesis shows

$$\llbracket c'_0 \rrbracket(a_1, a_2) \subseteq \llbracket \Phi \rrbracket$$

so by Lemma A.1.1, we conclude

$$\pi_1(\theta^{(i)}(a_1, a_2)) \leq \eta_1(a_1).$$

Subcase for Eq. (A.12): p_1 is true. If p_1 is true, then e_1 is true in a_1 by Eq. (A.4). Unfolding:

$$\theta^{(i)}(a_1, a_2) = \mathbf{bind}(\llbracket c'_1 \rrbracket(a_1, a_2), \theta^{(i-1)})$$
$$\eta_1(a_1) = \mathbf{bind}(\llbracket \mathbf{if}\ e_1\ \mathbf{then}\ c_1 \rrbracket a_1, \eta_1) = \mathbf{bind}(\llbracket c_1 \rrbracket a_1, \eta_1).$$

By the induction hypothesis, for every $(b_1, b_2) \in \llbracket \Phi \rrbracket$ we have $\pi_1(\theta^{(i-1)}(b_1, b_2)) \leq \eta_1(b_1)$. By the marginal condition from the outer induction hypothesis for the premise Eq. (A.9), we get

$$\pi_1(\llbracket c'_1 \rrbracket(a_1, a_2)) = \llbracket c_1 \rrbracket a_1.$$

Lemma A.1.1 establishes

$$\pi_1(\theta^{(i)}(a_1, a_2)) \leq \eta_1(a_1).$$

Subcase for Eq. (A.12): p_2 is true. If p_2 is true, then e_2 is true in a_2 by Eq. (A.5). Unfolding:

$$\theta^{(i)}(a_1, a_2) = \mathbf{bind}(\llbracket c'_2 \rrbracket(a_1, a_2), \theta^{(i-1)})$$
$$\eta_1(a_1) = \mathbf{bind}(\llbracket \mathbf{skip} \rrbracket a_1, \eta_1).$$

By the induction hypothesis, for every $(b_1, b_2) \in \llbracket \Phi \rrbracket$ we have $\pi_1(\theta^{(i-1)}(b_1, b_2)) \leq \eta_1(b_1)$. By the marginal condition from the outer induction on the premise Eq. (A.10), we get

$$\pi_1(\llbracket c'_2 \rrbracket(a_1, a_2)) = \llbracket \mathbf{skip} \rrbracket a_1.$$

Lemma A.1.1 establishes

$$\pi_1(\theta^{(i)}(a_1, a_2)) \leq \eta_1(a_1).$$

This completes the inductive case $i > 0$, establishing Eq. (A.12). By taking limits in Eqs. (A.11) and (A.12) and interchanging limits and projections (Lemma A.1.2), we have:

$$\pi_1(\theta_\times(a_1, a_2)) \leq \eta_1(a_1) \leq \pi_1(\theta_\times(a_1, a_2))$$

and hence equality holds, showing the first marginal condition.

The remaining equations Eqs. (A.13) and (A.14) for the second marginal condition follow by a symmetric argument, proving soundness of the rule.

This completes the induction, establishing soundness of ×PRHL.   □

Appendix B

Soundness of APRHL

The version of the logic APRHL we saw is similar to existing presentations of APRHL (cf. Barthe et al. (2013c); Barthe and Olmedo (2013); Olmedo (2014)). The main differences are our definition of approximate lifting (Definition 4.2.2), which is a variant of the approximate lifting introduced by Barthe and Olmedo (2013) and Olmedo (2014) with better theoretical properties, and the new proof rules introduced in Chapters 4 and 5.

We prove soundness of this version of APRHL, consisting of Figs. 4.1 to 4.5, 5.1, 5.4 and 5.6.

Theorem 4.3.2 (Soundness of APRHL). *Let ρ be a logical context. If a judgment is derivable*

$$\rho \vdash c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi,$$

then it is valid:

$$\rho \models c_1 \sim_{(\varepsilon,\delta)} c_2 : \Phi \Longrightarrow \Psi.$$

Proof. By induction on the height of the proof derivation. We consider the two-sided rules first (Fig. 4.1), followed by the one-sided rules (Fig. 4.2), and the structural rules (Fig. 4.3). The new rules (Figs. 4.4, 4.5, 5.1, 5.4 and 5.6) were proved sound in Chapters 4 and 5; we give pointers to the relevant lemmas.

If the premises are valid and we have two inputs m_1, m_2 that satisfy the pre-condition, we must construct witnesses μ_L, μ_R of the approximate lifting; namely, they must satisfy the support condition, the marginal conditions, and the distance condition in Definition 4.2.2. Let μ_1 and μ_2 be the output distributions from inputs m_1 and m_2 respectively. Throughout, we will leave the logical context ρ implicit when taking the semantics $[\![-]\!]$; these constants play no role in the proof.

Case [SKIP] Trivial; take $\mu_L = \mu_R = \mathbf{unit}(m_1, m_2)$.

Case [ASSN] Trivial; take $\mu_L = \mu_R = \mathbf{unit}(m_1[x_1 \mapsto v_1], m_2[x_2 \mapsto v_2])$ with $v_i \triangleq [\![e_i]\!] m_i$.

Case [LAP] Consequence of soundness for [LAPGEN] (Theorem 4.5.4)—in [LAPGEN], take $k \triangleq 0$ and $k' \triangleq k$ in [LAP].

Case [SEQ] By induction, we have two maps

$$\eta_L, \eta_R : \mathbf{State} \times \mathbf{State} \to \mathbf{Distr}(\mathbf{State}^\star \times \mathbf{State}^\star)$$

such that for any memories a_1, a_2 satisfying Φ, the distributions $\eta_L(a_1, a_2), \eta_R(a_1, a_2)$ witness the (ε, δ)-approximate lifting with support Ψ, and we have maps $\eta'_L, \eta'_R : \mathbf{State} \times \mathbf{State} \to \mathbf{Distr}(\mathbf{State}^\star \times \mathbf{State}^\star)$ such that for any memories a'_1, a'_2 satisfying Ψ, the distributions $\eta'_L(a'_1, a'_2), \eta'_R(a'_1, a'_2)$ witness the (ε', δ')-approximate lifting with support Θ.

To construct the witnesses for the conclusion, we would like to combine the witnesses for the premises in sequence. There is a slight mismatch, as $\eta_L(a_1, a_2)$ and $\eta_R(a_1, a_2)$ may place probability on pairs $(m, \star)$ or $(\star, m)$. Accordingly, we first extend the domain of the second maps η'_L, η'_R. We define

$$\widehat{\eta_L}(a'_1, a'_2)(x, y) \triangleq \eta'_L(a'_1, a'_2)(x, y) \quad \text{if} \quad a'_1, a'_2 \neq \star$$
$$\widehat{\eta_L}(a'_1, \star)(x, \star) \triangleq (\llbracket c'_1 \rrbracket a'_1)(x)$$
$$\widehat{\eta_L}(\star, a'_2)(\star, y) \triangleq (\llbracket c'_2 \rrbracket a'_2)(y)$$
$$\widehat{\eta_R}(a'_1, a'_2)(x, y) \triangleq \eta'_R(a'_1, a'_2)(x, y) \quad \text{if} \quad a'_1, a'_2 \neq \star$$
$$\widehat{\eta_R}(a'_1, \star)(x, \star) \triangleq (\llbracket c'_1 \rrbracket a'_1)(x)$$
$$\widehat{\eta_R}(\star, a'_2)(\star, y) \triangleq (\llbracket c'_2 \rrbracket a'_2)(y)$$

and zero otherwise. We now define the witnesses for the conclusion:

$$\mu_L \triangleq \mathbf{bind}(\eta_L(m_1, m_2), \widehat{\eta_L}) \quad \text{and} \quad \mu_R \triangleq \mathbf{bind}(\eta_R(m_1, m_2), \widehat{\eta_R}).$$

The support condition is clear, as

$$\mathrm{supp}(\eta_L), \mathrm{supp}(\eta_R) \subseteq \llbracket \Psi \rrbracket^\star \quad \text{and} \quad \mathrm{supp}(\widehat{\eta_L}(a_1, a_2)), \mathrm{supp}(\widehat{\eta_R}(a_1, a_2)) \subseteq \llbracket \Theta \rrbracket^\star$$

for all $a_1, a_2 \in \llbracket \Psi \rrbracket^\star$, by induction and by definition of $\widehat{\eta_L}, \widehat{\eta_R}$. The marginal conditions are also clear: by the marginal condition on η_L and η_R, we have $\eta_L(\star, a_2) = \eta_R(a_1, \star) = 0$ for all (a_1, a_2). Also note that for $a'_1 \neq \star$ we have

$$\pi_1(\widehat{\eta_L}(a'_1, a'_2)) = \llbracket c'_1 \rrbracket a'_1,$$

and for $a'_2 \neq \star$ we have

$$\pi_2(\widehat{\eta_R}(a'_1, a'_2)) = \llbracket c'_2 \rrbracket a'_2.$$

Therefore,

$$\begin{aligned}
\pi_1(\mu_L) &= \pi_1(\mathbf{bind}(\eta_L(m_1, m_2), \widehat{\eta_L})) \\
&= \mathbf{bind}(\pi_1(\eta_L(m_1, m_2)), \llbracket c'_1 \rrbracket) \\
&= \mathbf{bind}(\llbracket c_1 \rrbracket m_1, \llbracket c'_1 \rrbracket) \\
&= \llbracket c_1; c'_1 \rrbracket m_1
\end{aligned}$$

where the first equality is by Lemma A.1.1 and the marginal condition from the second premise, and the second equality is by the marginal condition from the first premise. For the second marginal,

$$\begin{aligned}
\pi_2(\mu_R) &= \pi_2(\mathbf{bind}(\eta_R(m_1, m_2), \widehat{\eta_R})) \\
&= \mathbf{bind}(\pi_2(\eta_R(m_1, m_2)), \llbracket c'_2 \rrbracket) \\
&= \mathbf{bind}(\llbracket c_2 \rrbracket m_2, \llbracket c'_2 \rrbracket) \\
&= \llbracket c_2; c'_2 \rrbracket m_2.
\end{aligned}$$

Thus, it only remains to check the distance condition $d_{\varepsilon+\varepsilon'}(\mu_L, \mu_R) \leq \delta + \delta'$. Let $\mathcal{S} \subseteq \mathbf{State}^\star \times \mathbf{State}^\star$ be any set of pairs memories, possibly including $\star$. We need to bound $\mu_L(\mathcal{S}) \leq \exp(\varepsilon) \cdot \mu_R(\mathcal{S}) + \delta$. Since $d_\varepsilon(\eta_L(m_1, m_2), \eta_R(m_1, m_2)) \leq \delta$, there exist constants $\zeta(x_1, x_2) \geq 0$ (possibly depending on m_1, m_2) for $x_1, x_2 \in \mathbf{State}^\star$ such that

$$\eta_L(m_1, m_2)(x_1, x_2) \leq \exp(\varepsilon) \cdot \eta_R(m_1, m_2)(x_1, x_2) + \zeta(x_1, x_2)$$

and

$$\sum_{(x_1, x_2) \in \textbf{State}^\star \times \textbf{State}^\star} \zeta(x_1, x_2) \le \delta.$$

By definition, for all $a_1', a_2' \in \textbf{State}^\star \times \textbf{State}^\star$ we have

$$d_{\varepsilon'}\left(\widehat{\eta_L}(a_1', a_2'), \widehat{\eta_R}(a_1', a_2')\right) \le \delta'.$$

Thus, we can directly compute (with all sums over $\textbf{State}^\star \times \textbf{State}^\star$):

$$
\begin{aligned}
\mu_L(\mathcal{S}) &= \sum_{(x_1, x_2)} \eta_L(m_1, m_2)(x_1, x_2) \cdot \widehat{\eta}_L(x_1, x_2)(\mathcal{S}) \\
&\le \sum_{(x_1, x_2)} \eta_L(m_1, m_2)(x_1, x_2) \cdot \min(\exp(\varepsilon')\widehat{\eta}_R(x_1, x_2)(\mathcal{S}) + \delta', 1) \\
&= \sum_{(x_1, x_2)} \eta_L(m_1, m_2)(x_1, x_2) \cdot (\min(\exp(\varepsilon')\widehat{\eta}_R(x_1, x_2)(\mathcal{S}), 1 - \delta') + \delta') \\
&= \delta' + \sum_{(x_1, x_2)} \eta_L(m_1, m_2)(x_1, x_2) \cdot \min(\exp(\varepsilon')\widehat{\eta}_R(x_1, x_2)(\mathcal{S}), 1 - \delta') \\
&\le \delta' + \sum_{(x_1, x_2)} (\exp(\varepsilon) \cdot \eta_R(m_1, m_2)(x_1, x_2) + \zeta(x_1, x_2)) \cdot \min(\exp(\varepsilon')\widehat{\eta}_R(x_1, x_2)(\mathcal{S}), 1 - \delta') \\
&\le \delta' + \sum_{(x_1, x_2)} \exp(\varepsilon) \cdot \eta_R(m_1, m_2)(x_1, x_2) \cdot \exp(\varepsilon')\widehat{\eta}_R(x_1, x_2)(\mathcal{S}) + \sum_{(x_1, x_2)} \zeta(x_1, x_2) \cdot (1 - \delta') \\
&\le \delta' + \sum_{(x_1, x_2)} \exp(\varepsilon) \cdot \eta_R(m_1, m_2)(x_1, x_2) \cdot \exp(\varepsilon')\widehat{\eta}_R(x_1, x_2)(\mathcal{S}) + (1 - \delta') \sum_{(x_1, x_2)} \zeta(x_1, x_2) \\
&\le \delta + \delta' + \exp(\varepsilon + \varepsilon') \sum_{(x_1, x_2)} \eta_R(m_1, m_2)(x_1, x_2) \cdot \widehat{\eta}_R(x_1, x_2)(\mathcal{S}) \\
&= \delta + \delta' + \exp(\varepsilon + \varepsilon')\mu_R(\mathcal{S}).
\end{aligned}
$$

This establishes the distance condition $d_{\varepsilon + \varepsilon'}(\mu_L, \mu_R) \le \delta + \delta'$. Thus, μ_L, μ_R are witnesses to the desired approximate lifting.

Case [COND] There are two cases. If e_1 is true in m_1, then e_2 is also true in m_2 by the pre-condition. Hence, $\llbracket \textbf{if } e_1 \textbf{ then } c_1 \textbf{ else } c_1' \rrbracket m_1 = \llbracket c_1 \rrbracket m_1$ and $\llbracket \textbf{if } e_2 \textbf{ then } c_2 \textbf{ else } c_2' \rrbracket m_2 = \llbracket c_2 \rrbracket m_2$, and we can take μ_L, μ_R to be the witnesses from the first inductive premise. Otherwise, if e_1 is false in m_1 then e_2 is false in m_2 and we take μ_L, μ_R to be the witnesses from the second inductive premise.

Case [WHILE] We prove that for every two memories $(a_1, a_2) \in \llbracket \Phi \rrbracket$, if $\llbracket e_v \rrbracket a_1 = k$ then we have

$$\llbracket \textbf{while } e_1 \textbf{ do } c_1 \rrbracket a_1 \; (\Phi \wedge \neg e_1 \langle 1 \rangle)^{\sharp(k \cdot \varepsilon, k \cdot \delta)} \; \llbracket \textbf{while } e_2 \textbf{ do } c_2 \rrbracket a_2.$$

The proof is by induction on k. In the base case $k = 0$, by the premises e_1 is false in a_1 and hence e_2 is false in a_2. Therefore, we have

$$\llbracket \textbf{skip} \rrbracket a_1 \; (\Phi \wedge \neg e_1 \langle 1 \rangle)^{\sharp(0,0)} \; \llbracket \textbf{skip} \rrbracket a_2$$

by taking witnesses $\eta_L = \eta_R \triangleq \textbf{unit}(a_1, a_2)$.

For the inductive step $k > 0$, if e_1 is false in a_1 then e_2 is false in a_2, both loops are equivalent to **skip** and we take the witnesses as in the base case. Otherwise, e_1 and e_2 are both true and we need to show

$$\llbracket c_1; \textbf{while } e_1 \textbf{ do } c_1 \rrbracket a_1 \; \Phi^{\sharp(k \cdot \varepsilon, k \cdot \delta)} \; \llbracket c_2; \textbf{while } e_2 \textbf{ do } c_2 \rrbracket a_2.$$

From the premise, for every two memories $(a_1, a_2) \in [\![\Phi]\!]$ with e_1 true in a_1, we have

$$[\![c_1]\!]a_1 \; (\Phi \wedge e_v \langle 1 \rangle < k)^{\sharp(\varepsilon, \delta)} \; [\![c_2]\!]a_2.$$

For every pair of memories b_1, b_2 satisfying Φ with $e_v < k$ in b_1, the induction hypothesis gives

$$[\![\textbf{while } e_1 \textbf{ do } c_1]\!]b_1 \; (\Phi \wedge \neg e_1 \langle 1 \rangle)^{\sharp((k-1)\cdot\varepsilon, (k-1)\cdot\delta)} \; [\![\textbf{while } e_2 \textbf{ do } c_2]\!]b_2.$$

Combining these two witnesses with the reasoning from the case for [SEQ], we have

$$[\![c_1; \textbf{while } e_1 \textbf{ do } c_1]\!]a_1 \; (\Phi \wedge \neg e_1 \langle 1 \rangle)^{\sharp(k\cdot\varepsilon, k\cdot\delta)} \; [\![c_2; \textbf{while } e_2 \textbf{ do } c_2]\!]a_2$$

as desired. Applying this claim for $a_1 \triangleq m_1, a_2 \triangleq m_2$ and $k \triangleq N$ establishes soundness of the rule.

Case [ASSN-L] ([ASSN-R] similar) Trivial; take $\mu_L = \mu_R = \textbf{unit}(m_1[x_1 \mapsto v_1], m_2)$ with $v_1 \triangleq [\![e_1]\!]m_1$.

Case [LAP-L] ([LAP-R] similar) Let $\lambda \in \textbf{Distr}(\mathbb{Z})$ be the distribution $[\![\textbf{Lap}_\varepsilon(e)]\!]m_1$. We define the witnesses

$$\mu_L(m_1[x_1 \mapsto v_1], m_2) = \mu_R(m_1[x_1 \mapsto v_1], m_2) \triangleq \lambda(v_1)$$

for every $v_1 \in \mathbb{Z}$, and zero otherwise. The support, marginal, and distance conditions are easy to check.

Case [COND-L] ([COND-R] similar) There are two cases. If e_1 is true in m_1, then

$$[\![\textbf{if } e_1 \textbf{ then } c_1 \textbf{ else } c_1']\!]m_1 = [\![c_1]\!]m_1.$$

We let μ_L, μ_R be the witnesses from the first premise by induction. Otherwise if e_1 is false in m_1, we let μ_L, μ_R be the witnesses from the second premise by induction.

Case [WHILE-L] ([WHILE-R] similar) Trivial; by soundness of the PRHL version using Proposition 4.2.5.

Case [CONSEQ] Trivial; take the witnesses from the premise by induction.

Case [EQUIV] Trivial; take the witnesses from the premise by induction.

Case [CASE] There are two cases. If $(m_1, m_2) \in [\![\Theta]\!]$, then the input memories satisfy the pre-condition in the first premise. Otherwise if $(m_1, m_2) \in [\![\neg\Theta]\!]$, then the input memories satisfy the pre-condition in the second premise. In either case, by induction we take the witnesses from the respective premise as the witnesses for the conclusion.

Case [TRANS] By Lemma 5.1.9.

Case [FRAME] By the induction hypothesis, there are witnesses μ_L', μ_R' to an (ε, δ)-approximate lifting of the two output distributions μ_1, μ_2 on inputs m_1, m_2. Let $V = \text{FV}(\Theta)$ be the free variables in Θ and suppose $m_1[V] = a_1$ and $m_2[V] = a_2$, where $m[V]: V \to \mathcal{V}$ is the restriction of m to V, and a_1, a_2 are maps $V \to \mathcal{V}$. Since c_1 and c_2 do not modify variables in V, memories m_1' in the support of μ_1 satisfy $m_1'[V] = a_1$ and memories m_2' in the support of μ_2 satisfy $m_2'[V] = a_2$.

By Proposition 4.2.6 and the inductive hypothesis, we can find witnesses μ_L, μ_R to an (ε, δ)-approximate lifting of μ_1, μ_2 such that

$$\text{supp}(\mu_L) \cup \text{supp}(\mu_R) \subseteq [\![\Psi]\!] \cap \{(m_1', m_2') \mid m_1'[V] = a_1, m_2'[V] = a_2\} \subseteq [\![\Psi \wedge \Theta]\!],$$

where the last inclusion holds because m_1, m_2 restricted to V satisfy Θ by assumption. Hence, μ_L, μ_R witness the desired approximate lifting.

Case [LapNull] By Theorem 4.5.2.

Case [LapGen] By Theorem 4.5.4.

Case [PW-Eq] By Theorem 4.6.2.

Cases [UtB-L] and [UtB-R] By Theorem 5.2.3.

Case [LapInt] By Theorem 5.3.5.

Case [While-AC] By Theorem 5.4.10. □

Bibliography

Ron Aharoni, Eli Berger, Agelos Georgakopoulos, Amitai Perlstein, and Philipp Sprüssel. 2011. The Max-Flow Min-Cut Theorem for Countable Networks. *Journal of Combinatorial Theory, Series B* 101, 1 (2011), 1–17. https://doi.org/10.1016/j.jctb.2010.08.002

Aws Albarghouthi and Justin Hsu. 2018. Synthesizing Coupling Proofs of Differential Privacy. *Proceedings of the ACM on Programming Languages* 1, POPL (2018). arXiv:cs.PL/1709.05361 http://arxiv.org/abs/1709.05361 To appear at ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Los Angeles, California.

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

Krzysztof R. Apt. 1981. Ten Years of Hoare's Logic: A Survey–Part I. *ACM Transactions on Programming Languages and Systems* 3, 4 (1981), 431–483. https://doi.org/10.1145/357146.357150

Krzysztof R. Apt. 1983. Ten years of Hoare's logic: A survey–Part II: Nondeterminism. *Theoretical Computer Science* 28, 1 (1983), 83–109. https://doi.org/10.1016/0304-3975(83)90066-X

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 arXiv:cs.LO/1503.04522

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 arXiv:cs.PL/1702.00374

Gilles Barthe, Juan Manuel Crespo, and César Kunz. 2011a. Relational Verification Using Product Programs. In *International Symposium on Formal Methods (FM), Limerick, Ireland (Lecture Notes in Computer Science)*, Vol. 6664. Springer-Verlag, 200–214. https://doi.org/10.1007/978-3-642-21437-0_17

Gilles Barthe, Juan Manuel Crespo, and César Kunz. 2013a. Beyond 2-Safety: Asymmetric Product Programs for Relational Program Verification. In *International Symposium on Logical Foundations of Computer Science (LFCS), San Diego, California (Lecture Notes in Computer Science)*, Vol. 7734. Springer-Verlag, 29–43. https://doi.org/10.1007/978--3-642--35722--0_3

Gilles Barthe, Pedro R. D'Argenio, and Tamara Rezk. 2011b. Secure Information Flow by Self-Composition. *Mathematical Structures in Computer Science* 21, 06 (2011), 1207–1252. https://doi.org/10.1017/S0960129511000193

Gilles Barthe, François Dupressoir, Benjamin Grégoire, César Kunz, Benedikt Schmidt, and Pierre-Yves Strub. 2013b. EasyCrypt: A Tutorial. In *Foundations of Security Analysis and Design (FOSAD) (Lecture Notes in Computer Science)*, Vol. 8604. Springer-Verlag, 146–166. https://doi.org/10.1007/978-3-319-10082-1_6 Tutorial Lectures.

Gilles Barthe, Thomas Espitau, Marco Gaboardi, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. 2017a. Formal Certification of Randomized Algorithms. (2017). http://justinh.su/files/papers/ellora.pdf

Gilles Barthe, Thomas Espitau, Benjamin Grégoire, Justin Hsu, Léo Stefanesco, and Pierre-Yves Strub. 2015a. Relational Reasoning via Probabilistic Coupling. In *International Conference on Logic for Programming, Artificial Intelligence and Reasoning (LPAR), Suva, Fiji (Lecture Notes in Computer Science)*, Vol. 9450. Springer-Verlag, 387–401. https://doi.org/10.1007/978-3-662-48899-7_27 arXiv:cs.LO/1509.03476

Gilles Barthe, Thomas Espitau, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. 2017b. Proving Uniformity and Independence by Self-Composition and Coupling. In *International Conference on Logic for Programming, Artificial Intelligence and Reasoning (LPAR), Maun, Botswana (EPiC Series in Computing)*, Vol. 46. 385–403. arXiv:cs.PL/1701.06477 https://arxiv.org/abs/1701.06477

Gilles Barthe, Thomas Espitau, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. 2018. Proving Expected Sensitivity of Probabilistic Programs. *Proceedings of the ACM on Programming Languages* 1, POPL (2018). arXiv:cs.PL/1708.02537 http://arxiv.org/abs/1708.02537 To appear at ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Los Angeles, California.

Gilles Barthe, Thomas Espitau, Justin Hsu, Tetsuya Sato, and Pierre-Yves Strub. 2017c. ⋆-Liftings for Differential Privacy. In *International Colloquium on Automata, Languages and Programming (ICALP), Warsaw, Poland (Leibniz International Proceedings in Informatics)*, Vol. 80. Schloss Dagstuhl–Leibniz Center for Informatics, 102:1–102:12. https://doi.org/10.4230/LIPIcs.ICALP.2017.102 arXiv:cs.LO/1705.00133

Gilles Barthe, Noémie Fong, Marco Gaboardi, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. 2016a. Advanced Probabilistic Couplings for Differential Privacy. In *ACM SIGSAC Conference on Computer and Communications Security (CCS), Vienna, Austria*. 55–67. https://doi.org/10.1145/2976749.2978391 arXiv:cs.LO/1606.07143

Gilles Barthe, Cédric Fournet, Benjamin Grégoire, Pierre-Yves Strub, Nikhil Swamy, and Santiago Zanella-Béguelin. 2014a. Probabilistic Relational Verification for Cryptographic Implementations. In *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), San Diego, California*. 193–206. https://doi.org/10.1145/2535838.2535847

Gilles Barthe, Marco Gaboardi, Emilio Jesús Gallego Arias, Justin Hsu, César Kunz, and Pierre-Yves Strub. 2014b. Proving Differential Privacy in Hoare Logic. In *IEEE Computer Security Foundations Symposium (CSF), Vienna, Austria*. 411–424. https://doi.org/10.1109/CSF.2014.36 arXiv:cs.LO/1407.2988

Gilles Barthe, Marco Gaboardi, Emilio Jesús Gallego Arias, Justin Hsu, Aaron Roth, and Pierre-Yves Strub. 2015b. Higher-Order Approximate Relational Refinement Types for Mechanism Design and Differential Privacy. In *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Mumbai, India*. 55–68. https://doi.org/10.1145/2676726.2677000 arXiv:cs.PL/1407.6845

Gilles Barthe, Marco Gaboardi, Emilio Jesús Gallego Arias, Justin Hsu, Aaron Roth, and Pierre-Yves Strub. 2016b. Computer-Aided Verification in Mechanism Design. In *Conference on Web and Internet Economics (WINE), Montréal, Québec (Lecture Notes in Computer Science)*, Vol. 10123. Springer-Verlag, 273–293. https://doi.org/10.1007/978-3-662-54110-4_20 arXiv:cs.GT/1502.04052

Gilles Barthe, Marco Gaboardi, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. 2016c. Proving Differential Privacy via Probabilistic Couplings. In *IEEE Symposium on Logic in Computer Science (LICS), New York, New York*. 749–758. https://doi.org/10.1145/2933575.2934554 arXiv:cs.LO/1601.05047

Gilles Barthe, Marco Gaboardi, Justin Hsu, and Benjamin C. Pierce. 2016d. Programming Language Techniques for Differential Privacy. *ACM SIGLOG News* 3, 1 (Jan. 2016), 34–53. https://doi.org/10.1145/2893582.2893591

Gilles Barthe, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. 2017d. Coupling Proofs Are Probabilistic Product Programs. In *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Paris, France*. 161–174. https://doi.org/10.1145/3009837.3009896 arXiv:cs.PL/1607.03455

Gilles Barthe, Benjamin Grégoire, and Santiago Zanella-Béguelin. 2009. Formal Certification of Code-Based Cryptographic Proofs. In *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Savannah, Georgia*. 90–101. https://doi.org/10.1145/1480881.1480894

Gilles Barthe, Boris Köpf, Federico Olmedo, and Santiago Zanella-Béguelin. 2013c. Probabilistic Relational Reasoning for Differential Privacy. *ACM Transactions on Programming Languages and Systems* 35, 3 (Nov. 2013), 9:1–9:49. https://doi.org/10.1145/2492061

Gilles Barthe and Federico Olmedo. 2013. Beyond Differential Privacy: Composition Theorems and Relational Logic for f-Divergences between Probabilistic Programs. In *International Colloquium on Automata, Languages and Programming (ICALP), Riga, Latvia (Lecture Notes in Computer Science)*, Vol. 7966. Springer-Verlag, 49–60. https://doi.org/10.1007/978-3-642-39212-2_8

Mihir Bellare and Phillip Rogaway. 2006. The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In *IACR International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT), Saint Petersburg, Russia (Lecture Notes in Computer Science)*, Vol. 4004. Springer-Verlag, 409–426. https://doi.org/10.1007/11761679_25

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. https://doi.org/10.1162/153244302760200704

Russ Bubley and Martin Dyer. 1997. Path Coupling: A Technique for Proving Rapid Mixing in Markov Chains. In *IEEE Symposium on Foundations of Computer Science (FOCS), Miami Beach, Florida*. 223–231. https://doi.org/10.1109/SFCS.1997.646111

Mads Buch. 2017. *Formalizing Differential Privacy*. Master's thesis. Aarhus University. http://madsbuch.com/thesis

Mark Bun and Thomas Steinke. 2016. Concentrated Differential Privacy: Simplifications, Extensions, and Lower Bounds. In *IACR Theory of Cryptography Conference (TCC), Beijing, China (Lecture Notes in Computer Science)*, Vol. 9985. Springer-Verlag, 635–658. https://doi.org/10.1007/978-3-662-53641-4_24 arXiv:cs.CR/1605.02065

Mark Bun, Thomas Steinke, and Jonathan Ullman. 2017. Make Up Your Mind: The Price of Online Queries in Differential Privacy. In *ACM–SIAM Symposium on Discrete Algorithms (SODA), Barcelona, Spain*. 1306–1325. https://doi.org/10.1137/1.9781611974782.85

David Callahan and Ken Kennedy. 1988. Compiling Programs for Distributed-Memory Multiprocessors. *The Journal of Supercomputing* 2, 2 (Oct. 1988), 151–169. https://doi.org/10.1007/BF00128175

Krishnendu Chatterjee, Hongfei Fu, and Amir Kafshdar Goharshady. 2016a. Termination Analysis of Probabilistic Programs through Positivstellensatz's. In *International Conference on Computer Aided Verification (CAV), Toronto, Ontario (Lecture Notes in Computer Science)*, Vol. 9779. Springer-Verlag, 3–22. https://doi.org/10.1007/978-3-319-41528-4_1

Krishnendu Chatterjee, Hongfei Fu, Petr Novotný, and Rouzbeh Hasheminezhad. 2016b. Algorithmic Analysis of Qualitative and Quantitative Termination Problems for Affine Probabilistic Programs. In *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Saint Petersburg, Florida*. 327–342. https://doi.org/10.1145/2837614.2837639

Krishnendu Chatterjee, Petr Novotný, and Đorđe Žikelić. 2017. Stochastic Invariants for Probabilistic Termination. In *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Paris, France*. 145–160. https://doi.org/10.1145/3009837.3009873

Yuxin Deng and Wenjie Du. 2009. The Kantorovich Metric in Computer Science: A Brief Survey. *Electronic Notes in Theoretical Computer Science* 253, 3 (2009), 73–82. https://doi.org/10.1016/j.entcs.2009.10.006

Yuxin Deng and Wenjie Du. 2011. *Logical, Metric, and Algorithmic Characterisations of Probabilistic Bisimulation*. Technical Report CMU-CS-11-110. Carnegie Mellon University. http://arxiv.org/abs/1103.4577

Josée Desharnais. 1999. *Labelled Markov Processes*. Ph.D. Dissertation. McGill University. www.collectionscanada.gc.ca/obj/s4/f2/dsk1/tape3/PQDD_0031/NQ64546.pdf

Josée Desharnais, Abbas Edalat, and Prakash Panangaden. 2002. Bisimulation for Labelled Markov Processes. *Information and Computation* 179, 2 (2002), 163–193. https://doi.org/10.1006/inco.2001.2962

Josée Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. 2003. Approximating Labelled Markov Processes. *Information and Computation* 184, 1 (2003), 160–200. https://doi.org/10.1016/S0890-5401(03)00051-8

Josée Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. 2004. Metrics for Labelled Markov Processes. *Theoretical Computer Science* 318, 3 (2004), 323–354. https://doi.org/10.1016/j.tcs.2003.09.013

Josée Desharnais, François Laviolette, and Mathieu Tracol. 2008. Approximate Analysis of Probabilistic Processes: Logic, Simulation and Games. In *International Conference on Quantitative Evaluation of Systems (QEST), Saint Malo, France*. IEEE, 264–273. https://doi.org/10.1109/QEST.2008.42

Edsger W. Dijkstra. 1976. *A Discipline of Programming*. Prentice Hall.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. 2006. Calibrating Noise to Sensitivity in Private Data Analysis. In *IACR Theory of Cryptography Conference (TCC), New York, New York (Lecture Notes in Computer Science)*, Vol. 3876. Springer-Verlag, 265–284. https://doi.org/10.1007/11681878_14

Cynthia Dwork and Aaron Roth. 2014. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends in Theoretical Computer Science* 9, 3–4 (2014), 211–407. https://doi.org/10.1561/0400000042

Cynthia Dwork, Guy N. Rothblum, and Salil Vadhan. 2010. Boosting and Differential Privacy. In *IEEE Symposium on Foundations of Computer Science (FOCS), Las Vegas, Nevada*. 51–60. https://doi.org/10.1109/FOCS.2010.12

Hamid Ebadi, Thibaud Antignac, and David Sands. 2016. Sampling and Partitioning for Differential Privacy. In *Conference on Privacy, Security and Trust (PST), Auckland, New Zealand*. IEEE, 664–673. https://doi.org/10.1109/PST.2016.7906954

Hamid Ebadi and David Sands. 2016. Featherweight PINQ. *Journal of Privacy and Confidentiality* 7, 2 (2016), 159–184. https://arxiv.org/abs/1505.02642

Hamid Ebadi, David Sands, and Gerardo Schneider. 2015. Differential Privacy: Now It's Getting Personal. In *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Mumbai, India*. 69–81. https://doi.org/10.1145/2676726.2677005

Luis María Ferrer Fioriti and Holger Hermanns. 2015. Probabilistic Termination: Soundness, Completeness, and Compositionality. In *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Mumbai, India*. 489–501. https://doi.org/10.1145/2676726.2677001

Nathanaël Fijalkow, Bartek Klin, and Prakash Panangaden. 2017. Expressiveness of Probabilistic Modal Logics, Revisited. In *International Colloquium on Automata, Languages and Programming (ICALP), Warsaw, Poland (Leibniz International Proceedings in Informatics)*, Vol. 80. Schloss Dagstuhl–Leibniz Center for Informatics, Dagstuhl, Germany, 105:1–105:12. https://doi.org/10.4230/LIPIcs.ICALP.2017.105

Robert W. Floyd. 1967. Assigning Meanings to Programs. In *Symposium on Applied Mathematics*. American Mathematical Society. https://doi.org/10.1007/978-94-011-1793-7_4

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

Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. 2012. Universally Utility-Maximizing Privacy Mechanisms. *SIAM Journal on Computing* 41, 6 (2012), 1673–1693. https://doi.org/10.1137/09076828X

Alessandro Giacalone, Chi-Chang Jou, and Scott A. Smolka. 1990. Algebraic Reasoning for Probabilistic Concurrent Systems. In *IFIP TC2 Working Conference on Programming Concepts and Methods*. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.56.3664

Robert Hall, Larry Wasserman, and Alessandro Rinaldo. 2013. Random Differential Privacy. *Journal of Privacy and Confidentiality* 4, 3 (2013). Issue 2. http://repository.cmu.edu/jpc/vol4/iss2/3

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 (Proceedings of Machine Learning Research)*, Vol. 48. 1225–1234. arXiv:cs.LG/1509.01240 http://arxiv.org/abs/1603.01445

Thomas P. Hayes and Eric Vigoda. 2007. Variable Length Path Coupling. *Random Structures and Algorithms* 31, 3 (2007), 251–272. https://doi.org/10.1002/rsa.20166

Charles A. R. Hoare. 1969. An Axiomatic Basis for Computer Programming. *Communications of the ACM* 12, 10 (1969), 576–580. https://doi.org/10.1145/363235.363259

Mark Jerrum. 1995. A Very Simple Algorithm for Estimating the Number of k-Colorings of a Low-Degree Graph. *Random Structures and Algorithms* 7, 2 (1995), 157–166. https://doi.org/10.1002/rsa.3240070205

Cliff B. Jones. 2003. The Early Search for Tractable Ways of Reasoning about Programs. *Annals of the History of Computing* 25, 2 (April 2003), 26–49. https://doi.org/10.1109/MAHC.2003.1203057

Bengt Jonsson and Kim Guldstrand Larsen. 1991. Specification and Refinement of Probabilistic Processes. In *IEEE Symposium on Logic in Computer Science (LICS), Amsterdam, The Netherlands*. 266–277. https://doi.org/10.1109/LICS.1991.151651

P. Kairouz, S. Oh, and P. Viswanath. 2017. The Composition Theorem for Differential Privacy. *IEEE Transactions on Information Theory* 63, 6 (June 2017), 4037–4049. https://doi.org/10.1109/TIT.2017.2685505

Shin-ya Katsumata and Tetsuya Sato. 2015. Codensity Liftings of Monads. In *International Conference on Concurrency Theory (CONCUR), Nijmegen, The Netherlands (Leibniz International Proceedings in Informatics)*, Vol. 35. Schloss Dagstuhl–Leibniz Center for Informatics, 156–170. https://doi.org/10.4230/LIPIcs.CALCO.2015.156

Jon Kleinberg and Eva Tardos. 2005. *Algorithm Design*. Addison-Wesley.

Dexter Kozen. 1981. Semantics of Probabilistic Programs. *Journal of Computer and System Sciences* 22, 3 (1981), 328–350. https://doi.org/10.1016/0022-0000(81)90036-2

Dexter Kozen. 1985. A Probabilistic PDL. *Journal of Computer and System Sciences* 30, 2 (1985). https://doi.org/10.1016/0022-0000(85)90012-1

V. S. Anil Kumar and H. Ramesh. 2001. Coupling vs. Conductance for the Jerrum-Sinclair Chain. *Random Structures and Algorithms* 18, 1 (2001), 1–17. http://hariharan-ramesh.com/papers/mcmc.pdf

Kim Guldstrand Larsen and Arne Skou. 1991. Bisimulation through Probabilistic Testing. *Information and Computation* 94, 1 (1991), 1–28. https://doi.org/10.1016/0890-5401(91)90030-6

David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. 2009. *Markov Chains and Mixing Times*. American Mathematical Society. http://pages.uoregon.edu/dlevin/MARKOV/markovmixing.pdf

Torgny Lindvall. 2002. *Lectures on the Coupling Method*. Courier Corporation.

Min Lyu, Dong Su, and Ninghui Li. 2017. Understanding the Sparse Vector Technique for Differential Privacy. *Proceedings of the VLDB Endowment* 10, 6 (2017), 637–648. https://doi.org/10.14778/3055330.3055331 Appeared at the International Conference on Very Large Data Bases (VLDB), Munich, Germany.

Annabelle McIver, Carroll Morgan, Benjamin Lucien Kaminski, and Joost-Pieter Katoen. 2018. A New Rule for Almost-Certain Termination. *Proceedings of the ACM on Programming Languages* 1, POPL (2018). https://arxiv.org/abs/1612.01091 To appear at ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Los Angeles, California.

Frank McSherry. 2009. Privacy Integrated Queries. In *ACM SIGMOD International Conference on Management of Data (SIGMOD), Providence, Rhode Island*. 19–30. https://doi.org/10.1145/1559845.1559850

Frank McSherry and Kunal Talwar. 2007. Mechanism Design via Differential Privacy. In *IEEE Symposium on Foundations of Computer Science (FOCS), Providence, Rhode Island*. 94–103. https://doi.org/10.1109/FOCS.2007.66

Ilya Mironov. 2017. Rényi Differential Privacy. In *IEEE Computer Security Foundations Symposium (CSF), Santa Barbara, California*. 263–275. https://doi.org/10.1109/CSF.2017.11

Carroll Morgan, Annabelle McIver, and Karen Seidel. 1996. Probabilistic Predicate Transformers. *ACM Transactions on Programming Languages and Systems* 18, 3 (1996), 325–353. https://doi.org/10.1145/229542.229547

Peter W. O'Hearn. 2007. Resources, Concurrency, and Local Reasoning. *Theoretical Computer Science* 375, 1 (2007), 271–307. https://doi.org/10.1016/j.tcs.2006.12.035 Festschrift for John C. Reynolds's 70th birthday.

Peter W. O'Hearn, John C. Reynolds, and Hongseok Yang. 2001. Local Reasoning about Programs That Alter Data Structures. In *International Workshop on Computer Science Logic (CSL), Paris, France (Lecture Notes in Computer Science)*, Vol. 2142. Springer-Verlag, 1–19. https://doi.org/10.1007/3-540-44802-0_1

Federico Olmedo. 2014. *Approximate Relational Reasoning for Probabilistic Programs*. Ph.D. Dissertation. Universidad Politécnica de Madrid. http://oa.upm.es/23088/1/FEDERICO_OLMEDO.pdf

Catuscia Palamidessi and Marco Stronati. 2012. Differential Privacy for Relational Algebra: Improving the Sensitivity Bounds via Constraint Systems. In *Workshop on Quantitative Aspects of Programming Languages (QAPL), Tallin, Estonia (Electronic Proceedings in Theoretical Computer Science)*, Vol. 85. Open Publishing Association, 92–105. https://doi.org/10.4204/EPTCS.85.7

James Gary Propp and David Bruce Wilson. 1996. Exact Sampling with Coupled Markov Chains and Applications to Statistical Mechanics. *Random Structures and Algorithms* 9, 1-2 (1996), 223–252. https://doi.org/10.1002/(SICI)1098-2418(199608/09)9:1/2<223::AID-RSA14>3.0.CO;2-O

Davide Proserpio, Sharon Goldberg, and Frank McSherry. 2014. Calibrating Data to Sensitivity in Private Data Analysis: A Platform for Differentially-Private Analysis of Weighted Datasets. *Proceedings of the VLDB Endowment* 7, 8 (2014), 637–648. https://doi.org/10.14778/2732296.2732300 Appeared at the International Conference on Very Large Data Bases (VLDB), Hangzhou, China.

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*. 157–168. https://doi.org/10.1145/1863543.1863568

John C. Reynolds. 2001. Intuitionistic Reasoning about Shared Mutable Data Structure. *Millennial Perspectives in Computer Science* 2, 1 (2001), 303–321. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.11.5999&rep=rep1&type=pdf

John C. Reynolds. 2002. Separation Logic: A Logic for Shared Mutable Data Structures. In *IEEE Symposium on Logic in Computer Science (LICS), Copenhagen, Denmark*. 55–74. https://doi.org/10.1109/LICS.2002.1029817

Ryan Rogers, Salil Vadhan, Aaron Roth, and Jonathan Ullman. 2016. Privacy Odometers and Filters: Pay-as-you-go Composition. In *Conference on Neural Information Processing Systems (NIPS), Barcelona, Spain*. 1921–1929. http://arxiv.org/abs/1605.08294

Halsey L. Royden and Patrick Fitzpatrick. 2010. *Real Analysis* (fourth ed.). Prentice Hall.

Walter Rudin. 1976. *Principles of Mathematical Analysis* (third ed.). McGraw-Hill.

Tetsuya Sato. 2016. Approximate Relational Hoare Logic for Continuous Random Samplings. In *Conference on the Mathematical Foundations of Programming Semantics (MFPS), Pittsburgh, Pennsylvania (Electronic Notes in Theoretical Computer Science)*, Vol. 325. Elsevier, 277–298. https://doi.org/10.1016/j.entcs.2016.09.043

Tetsuya Sato, Gilles Barthe, Marco Gaboradi, Justin Hsu, and Shin-ya Katsumata. 2017. Reasoning about Divergences for Relaxations of Differential Privacy. (2017). arXiv:cs.PL/1710.09010 http://arxiv.org/abs/1710.09010

Roberto Segala and Nancy A. Lynch. 1995. Probabilistic Simulations for Probabilistic Processes. *Nordic Journal of Computing* 2, 2 (1995), 250–273. https://doi.org/doi.org/10.1007/BFb0015027

Roberto Segala and Andrea Turrini. 2007. Approximated Computationally Bounded Simulation Relations for Probabilistic Automata. In *IEEE Computer Security Foundations Symposium (CSF), Venice, Italy*. 140–156. https://doi.org/10.1109/CSF.2007.8

Victor Shoup. 2004. Sequences of Games: A Tool for Taming Complexity in Security Proofs. Cryptology ePrint Archive, Report 2004/332. (2004). http://eprint.iacr.org/2004/332

Marcelo Sousa and Isil Dillig. 2016. Cartesian Hoare Logic for Verifying k-Safety Properties. In *ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), Santa Barbara, California*. 57–69. https://doi.org/10.1145/2908080.2908092

Volker Strassen. 1965. The Existence of Probability Measures with Given Marginals. *The Annals of Mathematical Statistics* (1965), 423–439. https://doi.org/10.1214/aoms/1177700153

Tachio Terauchi and Alex Aiken. 2005. Secure Information Flow as a Safety Problem. In *International Symposium on Static Analysis (SAS), London, England (Lecture Notes in Computer Science)*, Vol. 3672. Springer-Verlag, 352–367. https://doi.org/10.1007/11547662_24

Hermann Thorisson. 2000. *Coupling, Stationarity, and Regeneration*. Springer-Verlag.

Mathieu Tracol, Josée Desharnais, and Abir Zhioua. 2011. Computing Distances between Probabilistic Automata. In *Workshop on Quantitative Aspects of Programming Languages (QAPL), Saarbrücken, Germany (Electronic Proceedings in Theoretical Computer Science)*, Vol. 57. Open Publishing Association, 148–162. https://doi.org/10.4204/EPTCS.57.11

Michael Carl Tschantz, Dilsun Kaynar, and Anupam Datta. 2011. Formal Verification of Differential Privacy for Interactive Systems. In *Conference on the Mathematical Foundations of Programming Semantics (MFPS), Pittsburgh, Pennsylvania (Electronic Notes in Theoretical Computer Science)*, Vol. 276. Elsevier, 61–79. https://doi.org/10.1016/j.entcs.2011.09.015

Franck van Breugel and James Worrell. 2001a. An Algorithm for Quantitative Verification of Probabilistic Transition Systems. In *International Conference on Concurrency Theory (CONCUR), Aalborg, Denmark (Lecture Notes in Computer Science)*, Vol. 2154. Springer-Verlag, 336–350. https://doi.org/10.1007/3-540-44685-0_23

Franck van Breugel and James Worrell. 2001b. Towards Quantitative Verification of Probabilistic Transition Systems. In *International Colloquium on Automata, Languages and Programming (ICALP), Crete, Greece (Lecture Notes in Computer Science)*, Vol. 2076. Springer-Verlag, 421–432. https://doi.org/10.1007/3-540-48224-5_35

Cédric Villani. 2008. *Optimal Transport: Old and New*. Springer-Verlag. https://doi.org/10.1007/978-3-540-71050-9

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

Hongseok Yang. 2007. Relational Separation Logic. *Theoretical Computer Science* 375, 1 (2007), 308–334. https://doi.org/10.1016/j.tcs.2006.12.036 Festschrift for John C. Reynolds's 70th birthday.

Anna Zaks and Amir Pnueli. 2008. CoVaC: Compiler Validation by Program Analysis of the Cross-Product. In *International Symposium on Formal Methods (FM), Turku, Finland (Lecture Notes in Computer Science)*, Vol. 5014. Springer-Verlag, 35–51. https://doi.org/10.1007/978-3-540-68237-0_5

Danfeng Zhang and Daniel Kifer. 2017. LightDP: Towards Automating Differential Privacy Proofs. In *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL), Paris, France*. 888–901. https://doi.org/10.1145/3009837.3009884