Concepts of Computationally Efficient Samplability

by

Alex Edmonds

A thesis submitted in conformity with the requirements
for the degree of Master of Science
Graduate Department of Computer Science
University of Toronto

Abstract

Concepts of Computationally Efficient Samplability

Alex Edmonds
Master of Science
Graduate Department of Computer Science
University of Toronto
2017

This work considers a variety of notions of computationally efficient samplability. Introduced here for the first time is the sampling class *ExactPSamp*, consisting of those distributions from which samples may be produced efficiently by an inputless probabilistic machine where the accuracy of approximation, for samples of a given size, depends only on the amount of time given. This class will be contrasted with the previously considered more general class *PSamp*, where the accuracy of the sampling machine is allowed to depend on an input term. The task of providing evidence for the separation or collapse of these sampling classes is explored, in particular in relation to complexity theoretic assumptions. Also discussed are the monotonic properties of *ExactPSamp* which lead to the definition of intermediate classes and an alternative characterization of *ExactPSamp*. The sampling classes are related to their ensemble analogues and, finally, we consider the task of sampling uniformly from the certificates to an *NP* problem, with attention to completeness and the difficulty of using completeness to provide evidence for the separation of our sampling classes.

Acknowledgements

I owe my gratitude to a number of individuals who provided many interesting discussions on this topic. These include Toni Pitassi, Tomoyuki Yamakami, Russell Impagliazzo, Alistair Sinclair, Cameron Freer, Nate Ackerman and Jeff Edmonds. Special thanks go to Tomoyuki Yamakami who, in our personal communications, was the first to observe the monotonicity of *ExactPSamp* and define the class *MonoPSamp*, and who also provided invaluable comments on a draft of this work. Finally, I would like to thank my supervisor Dan Roy, for his essential guidance and support.

Contents

Chapter 1

Introduction

There are a variety of approaches to the representation of probability distributions by computation, in particular as it pertains to efficient sampling. Broadly speaking, computationally efficient sampling means the representation of a probability distribution by a probabilistic Turing machine for which the probability of producing a particular sample as output approximates the probability of the sample as determined by the given distribution. However, there are variety of approaches to the precise formulation of this general concept.

It is illustrative to consider the following example. A probabilistic Turing machine has access to independent uniformly random bits. In other words, any number of times, it may observe the outcome of a fair coin toss. Now suppose one wishes to use such a machine to simulate the result of an unfair coin toss where, with probability $\frac{1}{3}$, the result is 1, and, with probability $\frac{2}{3}$, the result is 0. If it is required that an output is produced after a fixed number k of random bits have been observed, then there are 2^k possible outcomes for the sequence of random bits. In particular, on a subset of these of size ℓ, the machine would return 1. On all other $2^k - \ell$ possible outcomes for the sequence of random bits, the machine would return 0. Hence, the probability of the output being 1 would be $\frac{\ell}{2^k}$. However, $\frac{\ell}{2^k}$ cannot equal $\frac{1}{3}$ for integers ℓ and k, since 2^k is not a multiple of 3. Thus, the machine cannot not sample exactly from the target distribution.

One possible way of resolving this issue is to provide the machine with an error ϵ as input and then require only that the probability that the machine produces 1 differs from $\frac{1}{3}$ by at most ϵ.

However, there is another approach to the computational simulation of an unfair $\frac{1}{3}$-coin that does not rely on an input term. Consider the algorithm which, on iteration k, observes the outcome of a pair of random bits. If the result is $(0, 0)$, then it halts and returns 1 as output. If the result is either $(0, 1)$ or $(1, 0)$, then it halts and returns 0 as output. However, if the result is $(1, 1)$, it proceeds to iteration $k + 1$. Then, for all k, conditioned on having halted on iteration k, the probability of producing 1 is $\frac{1}{3}$. Hence, the probability of ever producing 1 is also $\frac{1}{3}$, so that the machine samples from the $\frac{1}{3}$ coin exactly. Moreover, the probability of not halting within at most k steps is $\left(\frac{1}{4}\right)^k$, which decreases exponentially with k.

This last algorithm motivates the definition of the class *ExactPSamp* which is introduced here for the first time, and consists of those distributions from which samples may be produced efficiently by

an inputless machine for which the error of approximation depends only on the time it is given to run and the size of the sample produced. In particular, if the machine is allowed to run indefinitely, then the probability of it producing a sample is exactly the probability of the sample as determined by the target distribution. This will be contrasted with the class of distributions *PSamp*, as it consists of distributions that are samplable in a more general sense, where the error of approximation is determined by the machine's input.

The primary question which will be explored is whether *PSamp* and *ExactPSamp* are in fact distinct. Since *ExactPSamp* is a subset of *PSamp*, an equivalent question is whether the containment is proper. Towards addressing this question, we will examine some of the practical challenges to the task of translating from a machine witnessing a distribution in *PSamp* to a machine witnessing that the distribution is in *ExactPSamp*. Furthermore, we will consider the task of providing formal evidence for the separation of *PSamp* and *ExactPSamp*. However, separating these would imply $P \neq PP$, and thereby resolve a major open problem. For this reason, we consider the possibility of obtaining the separation of *PSamp* and *ExactPSamp* as a consequence of complexity theoretic assumptions, as well as some of the barriers to doing so.

Chapter 2

Basic definitions and notation

2.1 Strings and sets

Let $\mathbb{N}$ be the set of all nonnegative integers and let $\mathbb{R}$ be the set of all real numbers. For $n \in \mathbb{N}$, let $\{0,1\}^n$ denote the set of all binary strings of length n. Then let $\{0,1\}^*$ denote the set of all finite binary strings. Let $\{0,1\}^\infty$ denote the set of all infinite binary strings where the characters of the string are indexed over $\mathbb{N}$. If $\alpha \in \{0,1\}$, then, for $n \in \mathbb{N}$, let α^n denote the string made up of n repetitions of α. The symbol λ will represent the empty string. Moreover, define the relation $\sqsubseteq$, for $a, b \in \{0,1\}^* \cup \{0,1\}^\infty$, by $a \sqsubseteq b$ iff either $a = b$ or a is a proper prefix of b.

2.2 Turing machines

For either a deterministic or probabilistic Turing machine M, as usual let

$$M(x_1, \ldots, x_k)$$

denote its output given as input the tuple of strings $(x_1, \ldots, x_k)$ encoded in some canonical way. If M does not halt on input $(x_1, \ldots, x_k)$, then let

$$M(x_1, \ldots, x_k) = \perp$$

Using subscript, let

$$M_t(x_1, \ldots, x_k) = M(x_1, \ldots, x_k)$$

assuming the machine halts within t time steps on the given input. Otherwise, let $M_t(x_1, \ldots, x_k) = \perp$.

If M is a probabilistic Turing machine, then, at every time step, M observes a random bit. In particular, interpreting $b \in \{0,1\}^\infty$ as the outcome of an infinite sequence of random bits, let

$$M^b(x_1, \ldots, x_k)$$

denote the output of M on input $(x_1, \ldots, x_k)$ when acting with respect to b, so long as M halts then. Otherwise, let $M^b(x_1, \ldots, x_k) = \perp$. Extending the earlier notation, for $t \in \mathbb{N}$, let $M^b_t(x_1, \ldots, x_k) = M^b(x_1, \ldots, x_k)$ if M halts within t time steps on input $(x_1, \ldots, x_k)$ when acting with respect to b. Otherwise, let $M^b_t(x_1, \ldots, x_k) = \perp$.

Finally, using less conventional notation, for $a \in \{0,1\}^*$, let

$$M^a(x_1, \ldots, x_k) = y$$

if it holds that

$$\forall b \in \{0,1\}^\infty,\ a \sqsubseteq b \implies M^b(x_1, \ldots, x_k) = y$$

If no such y exists, then $M^a(x_1, \ldots, x_k) = \perp$. Similarly, let

$$M^a_t(x_1, \ldots, x_k) = y$$

if it holds that

$$\forall b \in \{0,1\}^\infty,\ a \sqsubseteq b \implies M^b_t(x_1, \ldots, x_k) = y$$

Again, if no such y exists, then $M^a_t(x_1, \ldots, x_k) = \perp$.

2.3 Probability theory

Typically, we consider probability distributions over the set $\{0,1\}^* \cup \{\perp\}$. For this purpose, it is usual to order the set $\{0,1\}^*$ of binary strings first by length and then lexicographically, with $x - 1$ denoting the predecessor of the string x according to this order. Then a (cumulative) distribution over $\{0,1\}^* \cup \{\perp\}$ is a nondecreasing function $\mu : \{0,1\}^* \to [0,1]$. Here, $\mu(x)$ is interpreted as the total probability of the occurrence of a sample less than or equal to x. Its associated density function $\hat{\mu} : \{0,1\}^* \cup \{\perp\} \to [0,1]$ is defined by

$$\hat{\mu}(x) = \begin{cases} \mu(\lambda), & \text{if } x = \lambda \\ \mu(x) - \mu(x-1), & \text{if } x \in \{0,1\}^* \setminus \{\lambda\} \\ 1 - \sup_{x \in \{0,1\}^*} \mu(x), & \text{if } x = \perp \end{cases}$$

Note that $\sum_{x \in \{0,1\}^*} \hat{\mu}(x) = 1$ is not required since positive probability may be assigned to the additional symbol $\perp$.

For a function $f : \{0,1\}^* \to \{0,1\}^*$, write $\mu_{f^{-1}}$ to denote the distribution with density function defined by

$$\hat{\mu}_{f^{-1}}(x) = \hat{\mu}(\{z \mid f(z) = x\})$$

Intuitively, it is useful to think of $\mu_{f^{-1}}$ as the distribution from which samples are produced by sampling first from μ and then applying the function f to the result.

Occasionally, it will be useful to consider distributions over $S \cup \{\perp\}$ where $S \subseteq \{0,1\}^*$. Thus we will refer to cumulative distributions of the form $\mu : S \to [0,1]$ with corresponding density function

of the form $\hat{\mu} : S \cup \{\perp\} \to [0,1]$. However, formally this may be represented as a distribution of the form $\mu : \{0,1\}^* \to [0,1]$ where the corresponding density function $\hat{\mu} : \{0,1\}^* \cup \{\perp\} \to [0,1]$ is supported on the set $S \cup \{\perp\}$, which is to say that $\hat{\mu}(x) = 0$ for $x \in \{0,1\}^* \setminus S$.

Chapter 3

Computational representation of distributions

3.1 Efficient computability

Before considering definitions of efficient samplability, it is worthwhile to introduce efficient computability, so that the former concept may be related to the latter.

A real number $b \in [0,1]$ may be expressed by its signed digit binary representation $b = \sum_{k=1}^{\infty} a_k \cdot 2^{-k}$, $a_k \in \{-1, 0, 1\}$, a representation which is considered computationally efficient if there is a Turing machine which, on input k, produces a_k in polynomial time of k. This somewhat unconventional representation is useful in a computational context where it guarantees closure under basic operations such as addition and multiplication, in contrast to standard binary representation which would not. Note that the existence of such a machine is equivalent to the existence of a machine which, on input k, produces an integer $M(k)$ satisfying $\left|b - 2^{-k} \cdot M(k)\right| \leq 2^{-k}$. Generalizing these ideas to functions leads to the following definition [5].

Definition 1. *Let $\mu : \{0,1\}^* \to \mathbb{R}$ be a distribution. Then say $\mu \in PComp$ (polynomial-time computable) if μ has a signed-digit representation $\mu(x) = \sum_{k=1}^{\infty} a_{x,k} \cdot 2^{-k}$, $a_{x,k} \in \{-1, 0, 1\}$, and there exists a polynomial-time Turing machine M such that $M(x, 0^k) = a_{x,k}$.*

One might ask why we consider efficient computability of only the cumulative distribution function μ and not that of the density function $\hat{\mu}$. This is because the efficient computability of $\hat{\mu}$ does not guarantee that samples may be generated efficiently from μ according to the definitions of the following section. For further discussion, refer to the work of Luca Trevisan [10].

3.2 Efficient samplability

The distribution of outputs of a probabilistic Turing machine M running in fixed time t has the property that, for any particular sample x, the probability that M produces x is an integer multiple

of 2^{-t}, known as a dyadic rational. This poses a severe limitation to the range of expression of a sampling machine running in fixed time, or even to more general machines where the running time is a function of only the size of the sample produced, such as those considered by Ben-David et al. [3]. In particular, it is evident, since *PComp* includes distributions with probabilities that are not dyadic, that these cannot be sampled exactly by a machine of the type just described.

This has caused a number of authors to consider a definition of efficient samplability where the sampling machine takes an accuracy parameter determining the error by which the probabilities of the machine's outputs are allowed to differ from the probabilities given by the original distribution. Paradigmatic of these definitions is the following [12].

Definition 2. *Let $\mu : \{0,1\}^* \to \mathbb{R}$ be a distribution. Then say $\mu \in$ PSamp if there exists a polynomial p and a probabilistic Turing machine M such that, for all $i \in \mathbb{N}$, for all $x \in \{0,1\}^*$,*

$$\left|\hat{\mu}(x) - \Pr\left(M_{p(|x|,i)}(0^i) = x\right)\right| \leq 2^{-i}$$

When $\mu \in$ *PSamp*, then the pair, consisting of the machine M and the polynomial p which satisfy the previous definiton, will be said to witness $\mu \in$ *PSamp*. This terminology will be adopted throughout for similar definitions also.

Above it is noted the importance of allowing the running time of M to depend on the error of approximation. It is also worth emphasizing here that the running time required to obtain a particular error is allowed to depend on the size of the sample. In particular, until the machine runs for at least time $p(n, 0)$, there are no guarantees on the approximation of $\hat{\mu}(x)$ for $|x| = n$, and the probability of having produced a sample of size n may even be zero. This contrasts with definitions where the total variation between μ and the distribution of the output of our machine is bounded by a function of the input and the running time. Indeed, the latter concept is not useful in a computational context where the distributions are over infinite sets since a probabilistic Turing machine, having run for a finite amount of time, has at most finitely many outputs that may be produced with positive probability.

For sake of clarity, it is worth pointing out that there are no explicit requirements regarding the machine's approximation of $\hat{\mu}(\perp)$ and the sampling machine never produces $\perp$.

The definition of *PSamp* allows the machine a great deal of flexibility in that the sampling procedure is allowed to depend on the required accuracy. By contrast, the next definition does not provide the machine with an accuracy term as input, so that, for samples of a particular size, the accuracy depends only on how much time the machine is given to run.

By contrast, we introduce here the class *ExactPSamp*, consisting of distributions from which samples may be produced efficiently by a machine that does not take input. Rather, for samples of a particular size, the accuracy attained depends only on the time given.

Definition 3. *Let $\mu : \{0,1\}^* \to \mathbb{R}$ be a distribution. Then say $\mu \in$ ExactPSamp if there exists a polynomial p and an inputless probabilistic Turing machine M such that*

$$\left|\hat{\mu}(x) - \Pr\left(M_{p(|x|,i)} = x\right)\right| \leq 2^{-i}$$

Clearly, *ExactPSamp* $\subseteq$ *PSamp*, since a machine which is witness to $\mu \in$ *ExactPSamp* is a machine witness to $\mu \in$ *PSamp* that simply ignores its input. On the other hand, it is far from evident that *PSamp* $\subseteq$ *ExactPSamp*, a problem that will be the primary focus of this work.

Significantly, a machine M witnessing $\mu \in$ *ExactPSamp* satisfies

$$\begin{aligned}
&\lim_{i\to\infty} \left|\hat{\mu}(x) - \Pr(M_{p(|x|,i)} = x)\right| \leq \lim_{i\to\infty} 2^{-i} \\
\implies &\left|\hat{\mu}(x) - \lim_{i\to\infty} \Pr\left(M_{p(|x|,i)} = x\right)\right| \leq \lim_{i\to\infty} 2^{-i} \\
\implies &\left|\hat{\mu}(x) - \Pr\left(M = x\right)\right| \leq 0 \\
\implies &\Pr(M = x) = \hat{\mu}(x)
\end{aligned}$$

In other words, if the machine M is allowed to run indefinitely, then the probability of it producing a sample x is exactly $\hat{\mu}(x)$. It is by this property that the name of the class is justified. Furthermore,

$$\begin{aligned}
&\forall x \in \{0,1\}^*,\ \Pr(M = x) = \hat{\mu}(x) \\
\implies &\sum_{x\in\{0,1\}^*} \Pr(M = x) = \sum_{x\in\{0,1\}^*} \hat{\mu}(x) \\
\implies &1 - \sum_{x\in\{0,1\}^*} \Pr(M = x) = 1 - \sum_{x\in\{0,1\}^*} \hat{\mu}(x) \\
\implies &\Pr(M = \perp) = \hat{\mu}(\perp)
\end{aligned}$$

Hence, $\hat{\mu}(\perp)$ corresponds to the probability that M does not halt.

The last notion of efficient samplability to be introduced in this section relies on the concept of p-honesty [6] [12].

Definition 4. *A function $f : \{0,1\}^* \to \{0,1\}^*$ is p-honest if there exists a polynomial p such that $|x| \leq p(|f(x)|)$.*

Definition 5. *Let $\mu : \{0,1\}^* \to \mathbb{R}$ be a distribution. Then say $\mu \in$ IPSamp if there exists $\nu \in$ PComp and p-honest $f \in$ FP such that $\mu = \nu_{f^{-1}}$*

At first glance, this may appear to be an unusual definition of samplability. However, it is justified by the fact that samples from $\mu = \nu_{f^{-1}}$ may be generated by sampling first from ν and then applying the function f to the result. Having f be p-honest guarantees that samples from $\nu_{f^{-1}}$ correspond to samples from ν that are not too large. These ideas are discussed formally in the next section.

3.3 Class containments

We will show that

$$PComp \subseteq IPSamp \subseteq ExactPSamp \subseteq PSamp$$

Indeed, it is immediate from the definitions that *PComp* $\subseteq$ *IPSamp* and *ExactPSamp* $\subseteq$ *PSamp*. Furthermore, the following containment may be obtained by application of inverse sampling. It is an extension of the result *PComp* $\subseteq$ *PSamp*, due to Ben-David et al. [3].

Proposition 6. *PComp* $\subseteq$ *ExactPSamp*.

Proof. Let $\mu \in$ *PComp*. Then a sampling machine M for μ may be defined by considering its random bittape as the binary representation of a real number w. Then, by binary search, M finds $x \in \{0,1\}^*$ such that either: $x = \lambda$ and $w \leq \mu(\lambda)$; or, $x \neq \lambda$ and $\mu(x-1) < w \leq \mu(x)$.

In detail: M first determines $n = |x|$ by iterating over index $m \in \mathbb{N}$, at each iteration deciding $\mu(0^m) \leq w$ or $\mu(0^m) > w$ by inspecting as many digits of $\mu(0^m)$ and w as are needed. Upon discovering the least value m where $\mu(0^m) > w$, the procedure terminates. Then, since μ is nondecreasing and $\mu(0^{|x|}) \leq \mu(x) < \mu(0^{|x|+1})$, it holds that $n = m - 1$.

Subsequently, M performs a binary search on strings of length n to find x. Supposing that a prefix d of x has been decided, then it is necessary to extend d to a prefix of x of length $|d|+1$. To do so, compare as many digits as is necessary to decide either $\mu(d0111\ldots) < w$ or $\mu(d0111\ldots) \geq w$. Then, since μ is nondecreasing, $\mu(d0111\ldots) < w$ implies that $d1$ is a prefix of x, and $\mu(d0111\ldots) \geq w$ implies that $d0$ is a prefix of x.

To analyze the algorithm, consider the case where

$$\mu(x-1) + 2^{-i-1} \leq w \leq \mu(x) - 2^{-k-1}$$

for some $i \in \mathbb{N}$. Then, it also holds that, for all $x' \in \{0,1\}^*$, $|\mu(x') - w| \geq 2^{-i}$. A prefix of length $i+3$ of the signed digit binary representation determines a real number within an error of 2^{-i-3}. Hence, in comparing the pair of reals w and $\mu(x')$ which satisfy $|w - \mu(x')| \geq 2^{-i-1}$, it suffices, to observe only the first $i+3$ digits of each to decide which is the larger. The first $i+3$ digits of $\mu(x')$ may be enumerated in polynomial time in $|x'|$ and i, since $\mu \in$ *PComp*, and all necessary operations may computed in time polynomial in i. Hence, deciding which is the larger of w and $\mu(x')$ requires only time polynomial of $|x'|$ and i. Furthermore, the number of comparisons made is at most polynomial in $|x|$ and each involves a comparison between w and $\mu(x')$ for some x' where $|x'| \leq |x| + 1$. It follows that, in this case, the entire algorithm runs in time polynomial of $|x|$ and k. Let p be that polynomial.

Since the condition

$$\mu(x-1) + 2^{-i-1} \leq w \leq \mu(x) - 2^{-i-1}$$

occurs with probability $\hat{\mu}(x) - 2^{-i}$, it holds that $\Pr(M_{p(|x|,i)} = x) \geq \hat{\mu}(x) - 2^{-i}$. Since the algorithm does not produce x unless $\mu(x-1) < w \leq \mu(x)$, it also holds that $\Pr(M_{p(|x|,k)} = x) \leq \hat{\mu}(x)$. Therefore,

$$\left|\Pr(M_{p(|x|,i)} = x) - \hat{\mu}(x)\right| \leq 2^{-i}$$ ■

The stronger containment *IPSamp* $\subseteq$ *ExactPSamp* may be obtained as a consequence of the closure properties of *ExactPSamp*.

Lemma 7. *If* $\nu \in$ *ExactPSamp and* $f \in$ *FP is p-honest, then* $\nu_{f^{-1}} \in$ *ExactPSamp.*

Proof. By generating a sample x from ν and returning the result of $f(x)$, one obtains a sample from $\nu_{f^{-1}}$. The condition of p-honesty guarantees that samples from $\nu_{f^{-1}}$ correspond to reasonably small samples from ν, lest generating a sample y from $\nu_{f^{-1}}$ requires waiting for a sample x from ν where $|x|$ is exponential in relation to $|y|$.

In detail: Let f be p-honest according to the polynomial q so that, for all $x \in \{0,1\}^*$, $|x| \leq q(|f(x)|)$. Also, let $\nu \in \mathit{ExactPSamp}$ be witnessed by machine M and polynomial p. Without loss of generality, assume that p and q are both nondecreasing. Then define $\overline{M}$ as the machine which simulates M until a sample x is produced and then returns $f(x)$.

Since, for all $t \in \mathbb{N}$,

$$\Pr(M_t = x) \leq \hat{\nu}(x)$$

it is easy to show that, for all $t \in \mathbb{N}$,

$$\Pr(\overline{M}_t = x) \leq \hat{\mu}(x)$$

Thus, it remains to provide the necessasry lower bound on $\Pr(\overline{M}_{p(|x|,i)} = x)$.

Let $i \in \mathbb{N}$. Then, for all $x \in \{0,1\}^*$,

$$\Pr(M_{p(|x|,i)} = x) \geq \hat{\nu}(x) - 2^{-i}$$

Since $\Pr(M_t = x)$ is nondecreasing with respect to t, $|x| \leq q(|f(x)|)$ implies

$$\Pr(M_{p(q(|f(x)|),i)} = x) \geq \hat{\nu}(x) - 2^{-i}$$

For an arbitrary string y, taking the sum of this inequality over $x \in f^{-1}(y)$ yields

$$\sum_{x \in f^{-1}(y)} \Pr(M_{p(q(|y|),i)} = x) \geq \sum_{x \in f^{-1}(y)} \hat{\nu}(x) - 2^{-i} \cdot \left|f^{-1}(y)\right| \geq \sum_{x \in f^{-1}(y)} \hat{\nu}(x) - 2^{-i+q(|y|)}$$

Then, by substitution of

$$\hat{\nu}_{f^{-1}}(y) = \sum_{x \in f^{-1}(y)} \hat{\nu}(x)$$

and

$$\Pr(f(M_{p(q(|y|),i)})) = y) = \sum_{x \in f^{-1}(y)} \Pr(M_{p(q(|y|),i)} = x)$$

we obtain

$$\Pr(f(M_{p(q(|y|),i)})) = y) \geq \hat{\nu}_{f^{-1}}(y) - 2^{-i+q(|y|)}$$

Conditioned on the event that $M_{p(q(|y|),i))} = y$, then the simulation of $M_{p(q(|y|),i))}$ together with the computation of $y = f(x)$ is achieved within time $p'(|y|, i)$ where p' is some fixed polynomial. Hence,

$$\Pr(\overline{M}_{p'(|y|,i)} = y) \geq \Pr\left(f(M_{p(q(|y|),i)}) = y\right)$$

which implies

$$\Pr(\overline{M}_{p'(|y|,i)} = y) \geq \hat{\nu}_{f^{-1}}(y) - 2^{-i+q(|y|)}$$

Finally, taking

$$p''(|y|, j) = p'(|y|, q(|y|) + j)$$

gives the desired result

$$\Pr(\overline{M}_{p''(|y|,j)} = y) \geq \hat{\nu}_{f^{-1}}(y) - 2^{-j}$$

■

Proposition 8. *IPSamp* $\subseteq$ *ExactPSamp.*

Proof. Suppose that $\mu \in$ *IPSamp* is witnessed by distribution $\nu \in$ *PComp* and p-honest function $f \in$ *FP* so that $\mu = \nu_{f^{-1}}$. By Proposition 6, $\nu \in$ *ExactPSamp*, and then, by Lemma 7, we have $\mu = \nu_{f^{-1}} \in$ *ExactPSamp*. ■

Chapter 4

Hardness

Recall that PP is the class of decision problems $\mathcal{L} \subseteq \{0,1\}^*$ for which there exists a polynomial time probabilistic machine M satisfying, for all $x \in \{0,1\}^*$,

$$x \in \mathcal{L} \text{ iff } \Pr(M(x) = 1) > \frac{1}{2}$$

As it turns out, the separation of $PComp$ from any one of our sampling classes is equivalent to the separation of P and PP. The equivalence of $P = PP$ and $PComp = PSamp$ is due to Miltersen [9]. That this result could be adapted to show the equivalence of $P = PP$ and a number of other statements, including $PComp = IPSamp$, was later noted by Yamakami [12].

Proposition 9. *The following statements are equivalent.*

1. $P = PP$
2. $PComp = IPSamp$
3. $PComp = ExactPSamp$
4. $PComp = PSamp$

Proof.

(A) $\underline{PComp = IPSamp \implies P = PP}$: Assume $PComp = IPSamp$. Let M be a polynomial-time probabilistic Turing machine witnessing $\mathcal{L} \in PP$. In particular, for all $x \in \{0,1\}^*$, $x \in \mathcal{L}$ iff $\Pr(M(x) = 1) > \frac{1}{2}$. Also, there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ such that, for all $x \in \{0,1\}^*$, the running time of M on input x is at most $p(|x|)$. We will make use of the encoding $\gamma : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ where $\gamma(x,b)$ is the concatenation of the strings $x_1 0 x_2 0 \ldots x_{|x|} 0$, 11, and b. Although not strictly a pairing function since it is not surjective, γ has the useful property that $\gamma(x,b) < \gamma(x',b')$ implies both $x < x'$ and $b < b'$.

Now consider the distribution μ defined by the following sampling procedure:

i. Sample $x \in \{0,1\}^*$ with probability $2^{-2|x|}$;

ii. Sample b uniformly from $\{0,1\}^{p(|x|)}$;

iii. Run $M^b(x)$;

iv. If $M^b(x) = 1$, then return $1x$. Otherwise, return $0x$.

To see that $\mu \in \mathit{IPSamp}$, let ν be the distribution which, for $x \in \{0,1\}^*$ and $b \in \{0,1\}^{p(|x|)}$, satisfies $\hat{\nu}(\gamma(x,b)) = 2^{-2|x|-p(|x|)}$. If $x \in \{0,1\}^*$ but $b \notin \{0,1\}^{p(|x|)}$, then $\hat{\nu}(\gamma(x,b)) = 0$. Also, if $y \in \{0,1\}^*$ is not a valid encoding of an ordered pair, which is to say $y \neq \gamma(x,b)$ for all $x, b \in \{0,1\}^*$, then $\hat{\nu}(x) = 0$. To see that $\nu \in \mathit{PComp}$, consider arbitrary $y \in \{0,1\}^*$. Let $x, b \in \{0,1\}^*$ maximize $\gamma(x,b)$ subject to the constraints $\gamma(x,b) \leq y$ and $|b| = p(|x|)$. Both x and b may be computed as a function of y in polynomial time. Then,

$$\begin{aligned}\nu(y) &= \nu(\gamma(x,b)) \\ &= \sum_{n<|x|} 2^{-2n-p(n)} + w(x) \cdot 2^{-2n-p(n)}\end{aligned}$$

where $w(x) = |\{x' \in \{0,1\}^{|x|} : x' \leq x\}|$. It holds that $w(x)$ may be computed in polynomial time of x since it is simply the integer whose binary representation is x. Since all other operations involved are also efficiently computable, it follows that $\nu \in \mathit{PComp}$.

It remains to define a p-honest function $f \in \mathit{FP}$ such that $\mu = \nu_{f^{-1}}$. To this end, for $x, b \in \{0,1\}^*$, $|b| = p(|x|)$, let $f(\gamma(x,b)) = M^b(x)$. If $y \neq \gamma(x,b)$ for all $x, b \in \{0,1\}^*$ satisfying $|b| = p(|x|)$, then let $f(y) = y$. It is immmediate that $f \in \mathit{FP}$. Moreover, since the sampling procedure for $\nu_{f^{-1}}$ which generates a sample from ν and applies f to the result evidently corresponds to the same distribution determined by the sampling procedure given for μ, it follows that $\mu = \nu_{f^{-1}}$.

Furthermore, when $x, b \in \{0,1\}^*$ satisfy $|b| = p(|x|)$, then $|f(\gamma(x,b))| = |x| + 1$ and $|\gamma(x,b)| = 2|x| + 2 + p(|x|)$ imply $|\gamma(x,b)| \leq q(|f(\gamma(x,b))|)$, where q is the polynomial defined by $p'(m) = 2m + p(m-1)$. When $y \neq \gamma(x,b)$ for all $x, b \in \{0,1\}^*$ satisfying $|b| = p(|x|)$, then $|y| = |f(y)|$ implies $|y| \leq q(|f(y)|)$. In short, for all $y \in \{0,1\}^*$, $|y| \leq q(|f(y)|)$. Hence f is p-honest.

Therefore, $\mu \in \mathit{IPSamp}$ so that, by consequence of the assumption that $\mathit{PComp} = \mathit{IPSamp}$, it also holds that $\mu \in \mathit{PComp}$.

To decide, for arbitrary $x \in \{0,1\}^*$, whether $x \in \mathcal{L}$, it suffices to decide $\mu(1x) > \mu(0x)$. Since $\mu \in \mathit{PComp}$, the first k digits of both $\mu(1x) > \mu(0x)$ may be enumerated in time polynomial of $|x|$ and k. Thus, it suffices to verify that the number of digits of $\mu(1x)$ and $\mu(0x)$ required to decide $\mu(1x) > \mu(0x)$ is polynomial in $|x|$. Indeed, since $M(x)$ runs in time $p(|x|)$, the output of M depends on the outcome of $p(|x|)$ random bits. It follows that $\Pr(M(x) = 1) = k \cdot 2^{-p(|x|)}$ for some $k \in \mathbb{N}$. Likewise, $\Pr(M(x) = 0) = k' \cdot 2^{-p(|x|)}$ for some $k' \in \mathbb{N}$. Hence

$$\Pr(M(x) = 1) > \Pr(M(x) = 0)$$

is equivalent to

$$\Pr(M(x) = 1) - \Pr(M(x) = 0) > 2^{-p(|x|)}$$

In turn, this is equivalent to

$$\mu(x1) - \mu(x0) > 2^{-p(|x|)-2|x|}$$

Thus, the first $p(|x|) + 2|x| + 2$ digits of the signed digit binary representations of $\mu(x0)$ and $\mu(x1)$, which determine their values up to an error of $2^{-p(|x|)-2|x|-1}$, are sufficient to decide $\mu(x1) > \mu(x0)$. Therefore, $\mathcal{L} \in P$.

(B) $P = PP \implies PComp = PSamp$: Assume $P = PP$. Let $\mu \in PSamp$ be witnessed by the machine M and the polynomial p. Then define a probabilistic Turing machine M' which takes as input: an accuracy term 0^i; a string $e[r]$, interpreted as the truncated binary representation of a dyadic real $r \in [0, 1]$; and a string $x \in \{0, 1\}^*$. On this input, M' executes either of the following processes with equal probability.

- Compute $M(0^i)$. If $M(0^i) > x$, then return 1. Otherwise, return 0.
- Sample 0 with probability r and 1 with probability $1 - r$.

Then M' has the important property that the probability of $M'(0^i, e[r], x) = 1$ is strictly greater than $\frac{1}{2}$ if and only if the probability of $M(0^i) > x$ is strictly greater than r. Moreover, since $P = PP$, there exists a deterministic machine M'' such that $M'(0^i, e[r], x) > \frac{1}{2}$ if and only if $M''(0^i, e[r], x) = 1$.

This allows us to recursively define a machine $\overline{M}$ where $\overline{M}(0^i, x)$ produces the ith digit of a fixed signed-digit binary representation of $\mu(x)$. In particular, given $y_i = \sum_{j=0}^{i} \overline{M}(0^j, x)$ which satisfies $|\mu(x) - y_i| \leq 2^{-i}$, then $\overline{M}(0^{i+1}, x)$ is computed by evaluating $M''(0^{i+2}, e[r], x)$ for $r = y_i - 2^{-i-2}$ and $r = y_i + 2^{-i-2}$. The individual cases are covered as follows.

Case 1: If $M''(0^{i+2}, e[y_i - 2^{-i-2}], x) = 0$, then $\Pr(M(0^{i+2}) = x) \leq y_i - 2^{-i-2}$. Hence,

$$\mu(x) \leq \Pr(M(0^{i+2}) = x) + 2^{-i-2} \leq y_i$$

Furthermore, by the precondition, $\mu(x) \geq y_i - 2^{-i-1}$. Thus by taking $\overline{M}(0^{i+1}, x)$ to be -1 it holds that

$$y_{i+1} := \sum_{j=0}^{i+1} \overline{M}(0^j, x)$$

satisfies $|\mu(x) - y_{i+1}| \leq 2^{-i}$.

Case 2: If $M''(0^{i+2}, e[y_i - 2^{-i-2}], x) = 1$ and $M''(0^{i+2}, e[y_i + 2^{-i-2}], x) = 0$, then

$$y_i - 2^{-i-2} < \Pr(M(0^{i+2}) = x) \leq y_i + 2^{-i-2}$$

from which it follows that

$$\begin{aligned} & y_i - 2^{-i-1} \\ \leq\ & \Pr(M(0^{i+2}) = x) - 2^{-i-2} \\ \leq\ & \mu(x) \\ \leq\ & \Pr(M(0^{i+2}) = x) + 2^{-i-2} \\ \leq\ & y_i + 2^{-i-1} \end{aligned}$$

Thus, by taking $\overline{M}(0^{i+1}, x)$ to be 0 it holds that

$$y_{i+1} := \sum_{j=0}^{i+1} \overline{M}(0^j, x)$$

again satisfies $|\mu(x) - y_{i+1}| \leq 2^{-i}$.

Case 3: Finally, if $M''(0^{i+2}, e[y_i + 2^{-i-2}], x) = 1$, then an argument similar to that of Case 1 shows that it suffices to take $\overline{M}(0^{i+1}, x)$ to be 1. In particular, it holds then that

$$y_{i+1} := \sum_{j=0}^{i+1} \overline{M}(0^j, x)$$

satisfies $|\mu(x) - y_{i+1}| \leq 2^{-i}$.

Having covered each case, it may be concluded that the recursive procedure is valid. Indeed, by induction, for all i, the value $y_i = \sum_{j=0}^{i} \overline{M}(0^j, x)$ satisfies $|\mu(x) - y_i| \leq 2^{-i}$. Therefore,

$$\mu(x) = \sum_{i=0}^{\infty} a_{x,i} \cdot 2^{-i}$$

where $a_{x,i} := \overline{M}(0^i, x)$. Since this is true for arbitrary x, it follows that $\mu \in \mathit{PComp}$.

(C) It has been shown that $\mathit{PComp} = \mathit{IPSamp}$ implies $P = PP$. The contrapositive says that $P \neq PP$ implies $\mathit{PComp} \neq \mathit{IPSamp}$. By the containments

$$\mathit{PComp} \subseteq \mathit{IPSamp} \subseteq \mathit{ExactPSamp} \subseteq \mathit{PSamp}$$

it holds that $\mathit{PComp} \neq \mathit{IPSamp}$ implies both $\mathit{PComp} \neq \mathit{ExactPSamp}$ and $\mathit{PComp} \neq \mathit{PSamp}$. In short, $P \neq PP$ implies that PComp is not equal to any of IPSamp, $\mathit{ExactPSamp}$ or PSamp.

For the converse, it has already been shown that $P = PP$ implies $\mathit{PComp} = \mathit{PSamp}$. By the above containments, $\mathit{PComp} = \mathit{PSamp}$ implies $\mathit{PComp} = \mathit{IPSamp} = \mathit{ExactPSamp} = \mathit{PSamp}$. ■

Corollary 10. *$P = PP$ implies $\mathit{IPSamp} = \mathit{ExactPSamp} = \mathit{PSamp}$.*

By Proposition 9, progress on separating PComp from our sampling classes is inextricable from

the question of $P = PP$. Indeed, in light of the widely held assumption that $P \neq PP$, Proposition 9 provides evidence that none of our sampling classes are equal to *PComp*.

Corollary 10 says that the separation of any of our sampling classes, for instance $PSamp \neq ExactPSamp$, would imply $P \neq PP$ and thereby resolve a major open problem. For this reason it should not be expected that a separation of *PSamp* and *ExactPSamp* can be obtained in the absence of complexity theoretic assumptions. However, Corollary 10 only provides the implication in one direction, leaving little idea of what particular assumptions might be required to separate *PSamp* from *ExactPSamp*.

Chapter 5

Average-case complexity agnosticism

In consideration of the implications that the separation of *PSamp* and *ExactPSamp* might have, it is natural to look for areas in complexity theory where there is an explicit appeal to probability distributions on the space of finite strings. Thus we are motivated to consider the domain of average-case complexity.

In this section, we only consider distributions $\mu : \{0,1\}^* \to [0,1]$ satisfying $\hat{\mu}(\perp) = 0$. Let $\mathcal{T}$ be the class of all distributions of this form. Then, for $\mu \in \mathcal{T}$, a machine M is said to be polynomial-time on μ-average if the running time $t(x)$, of M executing on input x, satisfies

$$\sum_{x \in \{0,1\}^*} \hat{\mu}(x) \frac{t(x)^\epsilon}{|x|} < \infty$$

for some constant $\epsilon > 0$. Also, a decision problem $\mathcal{L} \in \{0,1\}^*$ is said to be polynomial-time decidable on μ-average if there exists a machine M, that is polynomial-time on μ-average, which decides $\mathcal{L}$. By extension, when $\mathcal{F} \subseteq \mathcal{T}$ is a class of distributions, then $P_{\mathcal{F}}$ consists of those decision problems $\mathcal{L} \subseteq \{0,1\}^*$ which, for all $\mu \in \mathcal{F}$, are polynomial-time decidable on μ-average.

The concept of p-domination is essential in this context. The distribution ν is said to p-dominate the distribution μ, written $\mu \preceq^P \nu$, if there exists a polynomial p such that $\hat{\mu}(x) \leq p(|x|)\hat{\nu}(x)$ for all $x \in \{0,1\}^*$. When $\nu \preceq^P \mu$ and $\mu \preceq^P \nu$, write $\nu \approx^P \mu$. Likewise, when $\mathcal{F}$ and $\mathcal{G}$ are both classes of distributions, write $\mathcal{F} \preceq^P \mathcal{G}$ if

$$\forall \mu \in \mathcal{F},\ \exists \nu \in \mathcal{G},\ \mu \preceq^P \nu$$

When $\mathcal{F} \preceq^P \mathcal{G}$ and $\mathcal{G} \preceq^P \mathcal{F}$, write $\mathcal{F} \approx^P \mathcal{G}$.

Since containment implies p-domination, it is immediate that *IPSamp* $\preceq^P$ *PSamp*. Yamakami shows that the inverse relation holds as well [12] .

Proposition 11. *PSamp* $\cap \mathcal{T} \preceq^P$ *IPSamp* $\cap \mathcal{T}$.

Corollary 12. *IPSamp* $\cap \mathcal{T} \approx^P$ *ExactPSamp* $\cap \mathcal{T} \approx^P$ *PSamp* $\cap \mathcal{T}$.

Proof. By Proposition 11 together with the containments $\mathit{IPSamp} \subseteq \mathit{ExactPSamp} \subseteq \mathit{PSamp}$. ■

The significance of p-domination is the following simple fact, originally used by Levin [8] to define problems that are average-case complete.

Proposition 13. *Suppose $\mu, \nu \in \mathcal{T}$ satisfy $\mu \preceq^P \nu$. Then any machine M that is polynomial-time on ν-average is also polynomial-time on μ-average.*

Corollary 14. *Suppose $\mathcal{F}, \mathcal{G} \subseteq \mathcal{T}$ satisfy $\mathcal{F} \preceq^P \mathcal{G}$. Then, $P_{\mathcal{F}} \subseteq P_{\mathcal{G}}$.*

Corollary 15. $P_{\mathit{IPSamp} \cap \mathcal{T}} = P_{\mathit{ExactPSamp} \cap \mathcal{T}} = P_{\mathit{PSamp} \cap \mathcal{T}}$.

Proof. By Corollary 12. ■

Corollary 11 illustrates one sense in which *IPSamp*, *ExactPSamp* and *PSamp* are all "close" to each other. By contrast, the following fact, also due to Yamakami [12], shows that, in the same sense, *PComp* is "far" from each of our sampling classes, predicated on the separation of P and NP.

Proposition 16. $P \neq NP \implies \mathit{PSamp} \cap \mathcal{T} \not\preceq^P \mathit{PComp} \cap \mathcal{T}$.

Whereas Proposition 13 guarantees that p-equivalence implies average-case agnosticism, it does not guarantee the converse. In particular, Proposition 16 does not permit any immediate conclusions in the context of average-case complexity.

This topic has been taken further by Impagliazzo and Levin [6], whose result addresses the relevance, in the context of average-case complexity, of the distinction between *PComp* and *PSamp*. Their discussion relies on the notion of average-case completeness. It is enough to know that the formalization of reduction from (L, μ) to (L', μ') guarantees that, if there exists a probabilistic machine M' which decides the problem L' in polynomial time on μ'-average, then there exists a probabilistic machine M which decides the problem L in polynomial time on μ-average. Levin had previously shown that there existed a problem $L' \in NP$ and a distribution $\mu' \in \mathit{PComp}$ such that, for all problems L, for all distributions $\mu \in \mathit{PComp}$, (L, μ) is reducible to (L', μ') [8]. In particular, if L' is decidable by a probabilistic machine in polynomial time on μ'-average, then, for every $L \in NP$, for every $\mu \in \mathit{PComp}$, L is decidable by a probabilistic machine in polynomial time on μ-average. The pair (L', μ') is said to be average-case complete with respect to the class of distributions *PComp*. The consequence, of the result due Impagliazzo and Levin, is that, any problem (L', μ'), where $L' \in NP$ and $\mu' \in \mathit{PComp}$, which is average-case complete with respect to *PComp*, is average-case complete with respect to *IPSamp* in the following sense. Every pair (L, μ), where $L \in NP$ and $\mu \in \mathit{IPSamp}$, is reducible to (L', μ'). Although this result is surprising, it does not give much reason to suppose that there does not exist a meaningful distinction between *PComp* and *IPSamp* in the context of average-case complexity. For instance, it remains a possibility that some $L \in NP$ is decidable in polynomial time on μ-average for all distributions $\mu \in \mathit{PComp}$, while, for some more exotic distribution $\nu \in \mathit{IPSamp}$, it holds that (L, ν) is average-case complete with respect to *IPSamp*. In particular, although Impaliazzo and Levin's result would then guarantee the existence of some $L' \in NP$ and some $\mu' \in \mathit{PComp}$ such that (L, ν) reduced to (L', ν'), it would still allow for the possibility of $L \neq L'$. For a hypothetical situation which is less elaborate, consider a

problem $L \in NP$ and a distribution $\nu \in IPSamp$ where L is not decidable in polynomial time on ν-average. Although L is not decidable in polynomial time on ν-average, it may be that (L, ν) is still not average-case complete with respect *IPSamp*. In the absence of completeness, the result of Impagliazzo and Levin bears no immediate consequeneces, and it remains possible that L is decidable on μ-average for all $\mu \in PComp$.

Chapter 6

The random bit-tape view

Distributions in the sampling classes *PSamp* and *ExactPSamp* are witnessed by probabilistic Turing machines whose outputs depend on the configuration of their random bit-tape. For an arbitrary probabilistic machine M, which takes inputs of the form 0^i, let

$$A[M,t,x,i] := \{b \in \{0,1\}^* \mid M_t^b(0^i) = x \text{ and } \forall b' \sqsubseteq b \; M^{b'}(0^i) = \perp\}$$

where $b' \sqsubseteq b$ is the relation satisfied iff either $b' = b$ or b' is a proper prefix of b. Thus, $A[M,t,x,i]$ consists only of strings b of length at most t where, for all strings a of length t satisfying $b \sqsubseteq a$, it holds that $M_t^a(0^i) = x$. Moreover, because the sets are prefix-free, it is possible to assign them probabilities according to the function Λ, given for prefix-free $S \subseteq \{0,1\}^*$ by

$$\Lambda(S) = \sum_{b \in S} 2^{-|b|}$$

In turn, this allows us to rephrase our sampling definitions. In particular,

$$\Lambda(A[M, p(x,i), x, i]) = \Pr(M_{p(|x|,i)}(0^i) = x)$$

Thus, for instance, a distribution μ is a member of *PSamp* iff there exists a machine M and a polynomial p such that

$$\forall x \in \{0,1\}^*, \; \forall i \in \mathbb{N}, \; |\hat{\mu}(x) - \Lambda(A[M, p(x,i), x, i])| \leq 2^{-i}$$

6.1 Monotonicity

Consider an arbitary probabilistic Turing machine M which does not take input. If M halts and produces the string x within t time steps when acting on a particular random bit-string $b \in \{0,1\}^\infty$, then trivially it is true that M, acting with respect to b, would produce the same output in $t' > t$ time-steps. It follows from this that $\Pr(M_t = x)$ is nondecreasing as a function of t. Thus, if a

distribution $\mu \in \mathit{ExactPSamp}$ is witnessed by the machine M and a polynomial p, assumed to be strictly increasing without loss of generality, then it holds that $\Pr(M_{p(|x|,i)} = x)$ is nondecreasing with respect to i. Indeed, it is implicit in the definition of *ExactPSamp* that the approximation comes from below, namely,

$$\forall i \in \mathbb{N},\ \forall x \in \{0,1\}^*,\ \Pr(M_{p(|x|,i)} = x) \leq \mu(x)$$

These ideas motivate the definition of an intermediate sampling class, originally proposed by Tomoyuki Yamakami in our personal communications.

Definition 17. *Let MonoPSamp $\subseteq$ PSamp be the set of distributions μ whose membership in PSamp is witnessed by a machine M and a polynomial p where*

$$\Pr(M_{p(|x|,i)}(0^i) = x) = \Lambda(A[M, p(x,i), x, i])$$

is nondecreasing as a function of i.

Thus, a machine which is witness to $\mu \in \mathit{MonoPSamp}$ is allowed to vary its procedure depending on the accuracy parameter given as input, but it is required to obey the monotonicity properties of a *ExactPSamp* witness.

It is evident that

$$\mathit{ExactPSamp} \subseteq \mathit{MonoPSamp} \subseteq \mathit{PSamp}$$

Hence, the separation of *MonoPSamp* from either of the classes *ExactPSamp* or *PSamp* would imply the separation of *ExactPSamp* and *PSamp*. In particular, although the separation of *ExactPSamp* and *MonoPSamp* appears to be as elusive as the separation of *ExactPSamp* and *PSamp*, the question of the separation of *MonoPSamp* and *PSamp* may provide some focus in addressing our main questions.

Naive attempts fail to transform the witness M of a distribution $\mu \in \mathit{PSamp}$ into a witness $\overline{M}$ of $\mu \in \mathit{MonoPSamp}$. One particular barrier here is that the probability of $M_{p(|x|,i)}(0^i) = x$ may be as much as 2^{-i} even if $\hat{\mu}(x) = 0$. By contrast, if $\overline{M}$ is to be a valid witness of $\mu \in \mathit{MonoPSamp}$, then $\hat{\mu}(x) = 0$ implies that the machine never produces x as output. At the same time, it is required by definition, for some polynomial p', that whenever $\hat{\mu}(x) \geq 2^{-i+1}$, then the probability of $\overline{M}_{p(|x|,i)} = x$ is at least 2^{-i}.

6.2 Consistent samplability

It is possible to take Yamakami's notion of monotonic sampling even further by requiring that, if the machine M, acting with respect to the random bit-tape configuration $b \in \{0,1\}^\infty$ on input 0^i, returns x in time $p(|x|, i)$, then the machine M, still acting with respect to b but now on input 0^{i+1}, will return x in time $p(|x|, i+1)$.

Definition 18. *Let ConsistentPSamp $\subseteq$ PSamp be the set of distributions μ whose membership in*

PSamp is witnessed by a machine M and polynomial p which satisfy,

$$\forall b \in \{0,1\}^\infty,\ \forall x \in \{0,1\}^*,\ \forall i \in \mathbb{N},\ M^b_{p(|x|,i)}(0^i) = x \implies M^b_{p(|x|,i+1)}(0^{i+1}) = x$$

An alternative characterization of *ConsistentPSamp* in terms of the sets $A[M,t,x,i]$ may be obtained by introducing a partial ordering of finite prefix-free sets.

Definition 19. *Let $A, B \subseteq \{0,1\}^*$ be prefix-free sets. Then define the partial ordering $\leq$ by $A \leq B$ iff, for all $a \in A$, there exists $b \in B$ such that $b \sqsubseteq a$.*

Proposition 20. *For a distribution $\mu : \{0,1\}^* \to \mathbb{R}$, it holds that $\mu \in$ ConsistentPSamp iff $\mu \in$ PSamp is witnessed by a machine M and a polynomial p which satisfy, for all $x \in \{0,1\}^*$, for all $i \in \mathbb{N}$,*

$$A[M, p(|x|,i), x, i] \leq A[M, p(|x|, i+1), x, i+1]$$

Proof. ($\Rightarrow$) Assume $\mu \in$ *ConsistentPSamp* is witnessed by the machine M and the polynomial p. Let $a \in A[M, p(|x|,i), x, i]$ so that $M^a_{p(|x|,i)}(0^i) = x \neq \perp$. Then, by definition, for all $b \in \{0,1\}^\infty$ where $a \sqsubseteq b$, it holds that $M^b_{p(|x|,i)}(0^i) = x$. It follows that there must exist some string $a' \sqsubseteq a$ of minimal length which satisfies, for all $b \in \{0,1\}^\infty$, $a' \sqsubseteq b$ implies $M^b_{p(|x|,i)}(0^i) = x$. Hence, $a' \in A[M, p(|x|, i+1), x, i+1]$. Since a is arbitrary, we may conclude that

$$A[M, p(|x|,i), x, i] \leq A[M, p(|x|, i+1), x, i+1]$$

($\Leftarrow$) To prove the converse, suppose that $\mu \in$ *PSamp* is witnessed by a machine M and a polynomial p which satisfy, for all $x \in \{0,1\}^*$, for all $i \in \mathbb{N}$,

$$A[M, p(|x|,i), x, i] \leq A[M, p(|x|, i+1), x, i+1]$$

If, for some $b \in \{0,1\}^\infty$, $x \in \{0,1\}^*$, and $i \in \mathbb{N}$, it holds that $M^b_{p(|x|,i)}(0^i) = x$, then there must exist some $a \in \{0,1\}^*$ of minimal length which satisfies $M^a_{p(|x|,i)}(0^i) = x$. Since $A[M, p(|x|,i), x, i] \leq A[M, p(|x|, i+1), x, i+1]$, it follows that there exists some $a' \in A[M, p(|x|, i+1), x, i+1]$ such that $a' \sqsubseteq a$. Moreover, $a' \in A[M, p(|x|, i+1), x, i+1]$ implies that a' is a minimal length string satisfying $M^{a'}_{p(|x|,i+1)}(0^i) = x$. Hence, $a' \sqsubseteq a \sqsubseteq b$ implies $M^b_{p(|x|,i+1)}(0^i) = x$. Since b, x, and i are arbitrary, we may conclude that $\mu \in$ *ConsistentPSamp*. ■

To see that *ConsistentPSamp* $\subseteq$ *MonoPSamp*, consider the consequence that our partial ordering of prefix-free sets bears with regards to the order of their measures.

Proposition 21. *Let $A, B \subseteq \{0,1\}^*$ be finite prefix-free sets. If $A \leq B$, then $\Lambda(A) \leq \Lambda(B)$.*

Proof. Assume $A \leq B$. For all $q \in \{0,1\}^*$, let $A_q = \{a \in A : q \sqsubseteq a\}$. Given $b, b' \in B$, and $a \in A$ satisfying $a \in A_b \cap A_{b'}$, then $b \sqsubseteq a$ and $b' \sqsubseteq a$ so either $b \sqsubseteq b'$ or $b' \sqsubseteq b$. Since B is prefix-free, this implies $b = b'$. Hence, the sets $\{A_b\}_{b \in B}$ are disjoint. Moreover, since $A \leq B$, for all $a \in A$, there exists some $b \in B$ such that $b \sqsubseteq a$ and hence $a \in A_b$. Therefore, $\{A_b\}_{b \in B}$ is a partitioning of the set

A. Consequently,

$$\Lambda(A) = \sum_{b \in B} \Lambda(A_b)$$

Also, for arbitrary $q \in \{0,1\}^*$, since A_q is prefix free and contains only strings prefixed by q, it may be shown that $\Lambda(A_q) \leq 2^{-|q|}$. Therefore, $\Lambda(A) \leq \sum_{b \in B} 2^{-|b|} = \Lambda(B)$.

■

Corollary 22. *ConsistentPSamp* $\subseteq$ *MonoPSamp.*

It is also evident that *ConsistentPSamp* contains *ExactPSamp* since a machine witnessing a distribution $\mu \in$ *ExactPSamp* is a witness to $\mu \in$ *ConsistentPSamp* which simply ignores its input. As it turns out, the converse holds as well.

Proposition 23. *ConsistentPSamp* = *ExactPSamp.*

Proof. Let $\mu \in$ *ConsistentPSamp* be witnessed by the machine M and the polynomial p so that

$$A[M, p(|x|, i), x, i] \leq A[M, p(|x|, i+1), x, i+1]$$

It will be useful to order the pairs $(n, i) \in \mathbb{N} \times \mathbb{N}$ first by $n + i$ and then by n. In other words, $(n, i) < (n', i')$ iff $n+i < n'+i'$ or both $n+i = n'+i'$ and $n < n'$. Let $\{(n_k, i_k)\}_{k \in \mathbb{N}}$ be the sequence of all pairs in $\mathbb{N} \times \mathbb{N}$ in increasing order according to this ordering.

We wish to define a machine $\overline{M}$ witnessing $\mu \in$ *ExactPSamp*. To this end, for a random bit-tape configuration $b \in \{0,1\}^\infty$, let $\overline{M^b}$ be defined by the algorithm which, on the kth iteration, computes $M^b_{p(n,i)}(0^i)$ for $(n, i) = (n_k, i_k)$. If $M^b_{p(n,i)}(0^i) \in \{0,1\}^n$, which is to say that a string of length n is produced, then our algorithm halts and outputs $M^b_{p(n,i)}(0^i)$. Otherwise, the algorithm continues to iterate.

Now supposing that the algorithm does not halt before reaching the ordered pair (n, i) and that $M^b_{p(n,i)}(0^i) \in \{0,1\}^n$, then the ouput of the algorithm agrees with $M^b_{p(n,i)}(0^i)$.

On the other hand, consider the case where $M^b_{p(n,i)}(0^i) \in \{0,1\}^n$ but the algorithm halts when acting on some earlier pair $(n', i') < (n, i)$. If $i \neq i'$, then, by the definition of *ConsistentPSamp*, $M^b_{p(n,i)}(0^i) \in \{0,1\}^n$ implies

$$M^b_{p(n,i)}(0^i) = M^b_{p(n,i')}(0^{i'})$$

Then $M^b_{p(n,i')}(0^{i'}) \neq \perp$ and $M^b_{p(n',i')}(0^{i'}) \neq \perp$ imply

$$M^b_{p(n,i')}(0^{i'}) = M^b_{p(n',i')}(0^{i'})$$

Hence,

$$M^b_{p(n,i)}(0^i) = M^b_{p(n',i')}(0^{i'})$$

The case where $i' = i$ and $n' < n$ may be easily dealt with, to show that then also $M^b_{p(n,i)}(0^i) = M^b_{p(n',i')}(0^{i'})$.

To summarize, if $M^b_{p(n,i)}(0^i) \in \{0,1\}^n$, then our algorithm either produces $M^b_{p(n,i)}(0^i)$ when acting on the pair (n, i) or it produces $M^b_{p(n,i)}(0^i)$ when acting on some earlier pair (n', i'). Thus,

taking $T(n,i) \in O(p(n,i)^2)$ to be the time required for our algorithm to execute on each of the pairs $(n',i') \leq (n,i)$, it holds that $M^b_{p(n,i)}(0^i) \in \{0,1\}^n$ implies $\overline{M}^b_{T(n,i)} = M^b_{p(n,i)}(0^i)$. Hence,

$$\Pr(\overline{M}^b_{T(|x|,i)} = x) = \Pr(M^b_{p(|x|,i)}(0^i) = x)$$

It remains to show that T is bounded by a polynomial. Indeed,

$$\begin{aligned} T(n,i) &= \sum_{(n',i')<(n,i)} O(p(n',i')) \\ &\leq \sum_{(n',i')<(n,i)} O(p(n+i,n+i)) \\ &\leq O((n+i)^2 \cdot p(n+i,n+i)) \end{aligned}$$

■

Chapter 7

Relationship to ensemble definitions

7.1 Numerically indexed ensembles

Our previous definitions consider only distributions over $\{0,1\}^*$. However, in many circumstances, it is useful instead to consider an ensemble of distributions $\{\mu_n\}_{n\in\mathbb{N}}$ where each μ_n is a distribution over strings of size n. A sampling machine for such an ensemble takes the index n as an input parameter and produces samples of size n according to the distribution μ_n.

Definition 24. *Let $\{\mu_n\}_{n\in\mathbb{N}}$ be an ensemble of distributions $\mu_n : \{0,1\}^n \to \mathbb{R}$. Then,*

1. *$\{\mu_n\}_{n\in\mathbb{N}} \in PComp_{\mathrm{NUM}}$ if $\{\mu_n\}_{n\in\mathbb{N}}$ has a signed-digit representation $\mu_{|x|}(x) = \sum_{k=1}^{\infty} a_{x,k} \cdot 2^{-k}$, $a_{x,k} \in \{-1,0,1\}$, and there exists a polynomial-time Turing machine M such that $M(x,0^k) = a_{x,k}$*
2. *$\{\mu_n\}_{n\in\mathbb{N}} \in ExactPSamp_{\mathrm{NUM}}$ if there exists a polynomial p and a probabilistic Turing machine M such that $\left|\hat{\mu}_{|x|}(x) - \Pr\left(M_{p(|x|,i)}(0^{|x|}) = x\right)\right| \leq 2^{-i}$*
3. *$\{\mu_n\}_{n\in\mathbb{N}} \in PSamp_{\mathrm{NUM}}$ if there exists a polynomial p and a probabilistic Turing machine M such that $\left|\hat{\mu}_{|x|}(x) - \Pr\left(M_{p(|x|,i)}(0^{|x|}, 0^i) = x\right)\right| \leq 2^{-i}$*

To transform an ensemble $\{\mu_n\}_{n\in\mathbb{N}}$ into a distribution over the set $\{0,1\}^*$ of all finite strings, we view each μ_n as a distribution on $\{0,1\}^n$ and then form the mixture $\mu = \sum c_n \mu_n$, where $\sum_{n\in\mathbb{N}} c_n = 1$. In particular, it is convenient to take c_n to be $2^{-2\mathrm{llog}(n)-1}$ which is bounded above and below by the reciprocals of polynomials. This property is important in that the probability of seeing a sample of size n remains reasonably high for large n.

Not surprisingly, this prior takes an ensemble in $PComp_{\mathrm{NUM}}$, $ExactPSamp_{\mathrm{NUM}}$ or $PSamp_{\mathrm{NUM}}$ to a distribution of the corresponding class *PComp*, *ExactPSamp* or *PSamp*, as illustrated by the following proof.

Proposition 25. *Let $\{\mu_n\}_{n\in\mathbb{N}} \in ExactPSamp_{\mathrm{NUM}}$. Then the distribution $\pi : \{0,1\}^* \to \mathbb{R}$ defined by $\hat{\pi}(x) = 2^{-2\mathrm{llog}(|x|)-1} \cdot \hat{\mu}_{|x|}(x)$ satisfies $\pi \in ExactPSamp$.*

Proof. Let $\{\mu_n\}_{n\in\mathbb{N}} \in \mathit{ExactPSamp}_{\mathrm{NUM}}$ be witnessed by a machine M and the polynomial p. Then let $\overline{M}$ be the machine that produces samples from π by way of the following procedure:

i. Sample $n = |x|$ with probability $2^{-2\mathrm{llog}(n)-1}$ as follows:

   (a) Sample $k = \mathrm{llog}(n)$, with probability 2^{-k-1}, in $O(k)$ times-steps, by taking k to be the number of coin flips observed up until first observing heads;

   (b) Sample n uniformly from the interval $[2^k - 1, 2^{k+1} - 1)$. To achieve this in $O(n)$ time, first sample $n+1$ from the interval $[2^k, 2^{k+1})$ by sampling each of its non-leading binary digits independently and uniformly. Then subtract by one to obtain n.

ii. Using the machine M, sample x from the distribution μ_n.

Then

$$\Pr\left(\overline{M}_{p(|x|,i)+O(|x|)} = x\right) = 2^{-2\mathrm{llog}(|x|)-1} \cdot \Pr\left(M_{p(|x|,i)}(0^{|x|}) = x\right)$$

so that

$$\left|\hat{\mu}_{|x|}(x) - \Pr\left(M_{p(|x|,i)} = x\right)\right| \leq 2^{-i}$$

implies

$$\left|\hat{\pi}(x) - \Pr\left(\overline{M}_{p(|x|,i)+O(|x|+\mathrm{llog}(|x|))} = x\right)\right| \leq 2^{-i-2\mathrm{llog}(|x|)-1} \leq 2^{-i}$$ ■

In the cases of *PSamp* and *PComp*, the converse also holds, as formulated in the following proposition. This enables any discussion of these classes to implicitly encompass the corresponding ensemble classes so that, for instance, the collapse $\mathit{PSamp} = \mathit{PComp}$ would imply the collapse $\mathit{PSamp}_{\mathrm{NUM}} = \mathit{PComp}_{\mathrm{NUM}}$.

Proposition 26. *Let $\{\mu_n\}_{n\in\mathbb{N}}$ be an ensemble of distributions of the form $\mu_n : \{0,1\}^n \to \mathbb{R}$. Let $\pi : \{0,1\}^* \to \mathbb{R}$ be the distribution defined by $\hat{\pi}(x) = 2^{-2\mathrm{llog}(|x|)-1} \cdot \hat{\mu}_{|x|}(x)$. Then,*

1. *$\{\mu_n\}_{n\in\mathbb{N}} \in \mathit{PComp}_{\mathrm{NUM}}$ iff $\pi \in \mathit{PComp}$*
2. *$\{\mu_n\}_{n\in\mathbb{N}} \in \mathit{PSamp}_{\mathrm{NUM}}$ iff $\pi \in \mathit{PSamp}$*

Proof.

1. By the fact that multiplication and division of signed-digit reals is polynomial-time computable.
2. ($\Rightarrow$) The proof is similar to that of Proposition 25.

   ($\Leftarrow$) Let $\pi \in \mathit{PSamp}$ be witnessed by the machine M and polynomial p. We will define a machine $\overline{M}$, takes inputs 0^n and 0^j denoting sample size and accuracy respectively, and executes $M(0^i)$ up to k times. As soon as a sample x of size n is obtained, it is returned as the output of $\overline{M}$. If no such sample is obtained, then M produces no output.

   It remains to show that i and k need only be polynomial in j and n to approximate $\hat{\mu}_n(x)$ with error at most 2^{-j}.

(A) First, we wish to lower bound the probability that $M(0^i)$ produces a sample of size n. This is obtained by taking the sum, over samples x of size n, of the lower bound provided by the definition of *PSamp*:

$$\begin{aligned} & \Pr\left(\left|M_{p(n,i)}(0^i)\right| = n\right) \\ = & \sum_{|x|=n} \Pr\left(M_{p(n,i)}(0^i) = x\right) \\ \geq & \sum_{|x|=n} \left(\pi(x) - 2^{-i}\right) \\ = & \ 2^{-2\mathrm{llog}(n)-1} - 2^{n-i} \end{aligned}$$

(B) It follows that, after k iterations of $M(0^i)$, the probability of not seeing a sample of size n is bounded above by

$$\begin{aligned} \Psi \ := & \ \left(1 - 2^{-2\mathrm{llog}(n)-1} + 2^{n-i}\right)^k \\ \leq & \ \left(1 - 2^{-2\mathrm{llog}(n)-1}\right)^k + 2^{n-i+k}, \quad \text{since } (\alpha+\beta)^k \leq \alpha^k + \beta \cdot 2^k \text{ for } \alpha, \beta \in [0,1] \\ \leq & \ e^{-k \cdot 2^{-2\mathrm{llog}(n)-1}} + 2^{n-i+k}, \qquad \text{since } 1 + \alpha \leq e^\alpha \\ \leq & \ 2^{-\frac{k}{2\cdot(n+1)^2}} + 2^{n-i+k} \end{aligned}$$

Note that the inequality $(\alpha+\beta)^k \leq \alpha^k + \beta \cdot 2^k$ is obtained as a consequence of the binomial theorem. Indeed, for $\alpha, \beta \in [0,1]$,

$$(\alpha+\beta)^k = \sum_{t=0}^{k} \binom{k}{t} \alpha^{k-t}\beta^t = \alpha^k + \sum_{t=1}^{k} \binom{k}{t} \alpha^{k-t}\beta^t \leq \alpha^k + \sum_{t=1}^{k} \binom{k}{t} \beta = \alpha^k + 2^k \cdot \beta$$

Now by taking k to be polynomial $p_1(n,j) := 2(j+2)(n+1)^2$ and taking i to be at least polynomial $p_2(n,j,k) := n+j+k+2$, we can guarantee that the probability of not seeing a sample of size n is at most 2^{-j-1}.

(C) Next we show that, conditioned on producing a sample of size n, the probability that M produces any sample x of size n approximates $\hat{\mu}(x)$. Let

$$\begin{aligned} A &:= \pi(x) & \Delta A &:= \Pr(M_{p(n,i)} = x) - A \\ B &:= \sum_{|y|=n} \pi(y) & \Delta B &:= \Pr(|M_{p(n,i)}| = |x|) - B \\ Q &:= \frac{A}{B} = \frac{\pi(x)}{\sum_{|y|=n} \pi(y)} = \hat{\mu}_n(x) & \Delta Q &:= \frac{\Pr(M_{p(n,i)} = x)}{\Pr(|M_{p(n,i)}| = |x|)} - Q \end{aligned}$$

Then, we may bound ΔQ as follows

$$\begin{aligned}
|\Delta Q| &= Q \cdot \frac{|(A+\Delta A)\cdot B - (B+\Delta B)\cdot A|}{A\cdot(B+\Delta B)} \\
&\leq 2Q \cdot \frac{|(A+\Delta A)\cdot B - (B+\Delta B)\cdot A|}{A\cdot B} && \text{if } |\Delta B| \leq B/2 \\
&= 2Q \cdot \left|\frac{\Delta A}{A} - \frac{\Delta B}{B}\right| \\
&\leq 2Q \cdot \left(\frac{\max(\Delta A, \Delta B)}{\min(A,B)}\right) \\
&\leq 2Q \cdot \left(\frac{2^{n-i}}{\pi(x)}\right) \\
&= 2^{n-i+1} \cdot \frac{\hat{\mu}_n(x)}{\pi(x)} && \text{since } Q = \hat{\mu}_n(x) \\
&= 2^{n-i+1} \cdot 2^{2\mathrm{llog}(n)+1} \\
&\leq 2^{3n-i+2}
\end{aligned}$$

By taking i to be at least polynomial $p_3(n,j) := j + 3n + 3$, we obtain $|\Delta Q| \leq 2^{-j-1}$.

(D) In short, it is enough to take k to be polynomial $p_1(n,j)$ and i to be polynomial $p_4(n,j) := p_2(n, p_1(n,j)) + p_3(n,j)$. Then,

- By part (B), with probability at least $1 - 2^{-j-1}$, executing $M(0^i)$ iteratively k times produces a sample of size n;
- By part (C), conditioned on having produced a sample of size n, the probability that $M(0^i)$ produces a particular sample x of size n is approximately $\hat{\mu}_n(x)$, within additive error 2^{-j-1}.

Furthermore, the time required for k iterations of $M(0^i)$ is $O(p_4(n,j))$ where $p_4(n,j) = p_1(n,j) \cdot p(n, p_4(n,j))$. Combining these results, we obtain

$$\hat{\mu}_n(x) - 2^{-i} \leq (1 - 2^{-i-1})(\hat{\mu}_n(x) - 2^{-i-1}) \leq \Pr(M_{p_4(n,j)}(0^n, 0^i) = x) \leq \hat{\mu}_n(x) + 2^{-i-1}$$

■

By contrast, for an ensemble $\{\mu_n\}_{n\in\mathbb{N}}$ and the corresponding distribution π over the set $\{0,1\}^*$ of all finite strings, it is not evident that $\pi \in \mathit{ExactPSamp}$ implies $\{\mu_n\}_{n\in\mathbb{N}} \in \mathit{ExactPSamp}_{\mathrm{NUM}}$. Supposing that $\pi \in \mathit{ExactPSamp}$ is witnessed by a machine M and polynomial p, it may be natural to define $\overline{M}(0^n)$ by the procedure which iteratively executes M, each time running M until halting, and continues iterating until observing a sample of size n. This is closely related to the strategy used in Proposition 26 to show that $\pi \in \mathit{ExactPSamp}$ implies $\{\mu_n\}_{n\in\mathbb{N}} \in \mathit{ExactPSamp}_{\mathrm{NUM}}$, the difference now being that the execution of M cannot be guaranteed to halt in a fixed amount time.

To see why this should not be expected to work in general, it suffices to consider the special case where M does not produce any sample x of size n before $p(n,0)$ time steps, where it is assumed that $p(n,0) \in \Omega(n)$. Then, for all $m > n$, the probability that $|M| \neq n$ and M does not halt in $p(m,0)$

time steps is at least

$$\sum_{|x|=m+1} \pi(x) = 2^{-2\text{llog}(m)-1}$$

which decreases only polynomially with respect to m. It follows that, in polynomial time, $\overline{M}$ approximates μ_n with at best polynomial error, rather than with exponential error as required.

To remedy the shortcomings of our machine $\overline{M}$, it is tempting to modify $\overline{M}$ so that the simulation of M is halted in a predetermined amount time. Unfortunately, at best, this type of strategy results in a sampling machine of the *PSamp* rather than *ExactPSamp* type. Variations of this strategy, which place greater probability on halting M early, simply skew the probability of those samples that would be produced later on.

Another strategy is to adjust the prior which places a probability of $2^{-\text{llog}(n)-1}$ on samples of length n. However, here there is a trade-off between the halting time of the machine M witnessing $\pi \in$ *ExactPSamp* versus the number of iterations of M required to observe a sample of size n. For instance, if the probability of length n samples was taken to be 2^{-n-1}, one could guarantee that the probability of M not halting would decrease exponentially over time. However, then the expected number of iterations of $\overline{M}$ required to observe a sample of size n would be $2^{O(n)}$.

With these ideas in mind, the question of whether $\pi \in \textit{ExactPSamp}_{\text{NUM}}$ implies $\{\mu_n\}_{n\in\mathbb{N}} \in$ *ExactPSamp* appears to present many of the same challenges as the main question of *ExactPSamp* = *PSamp*.

7.2 String-indexed ensembles

Certain conceptions of ensembles are not accounted for by our previous definitions. Indeed, it is sometimes useful to allow the distributions to be indexed by arbitrary strings. Thus, consider the ensemble $\{\mu_x\}_{x\in\{0,1\}^*}$ where, for some strictly increasing polynomial q, each μ_x is a distribution over strings of length $p(|x|)$. Then a sampling machine for such an ensemble takes the string x as an input parameter and produces samples of size $p(|x|)$ according to the distribution μ_x.

Definition 27. *Let $\{\mu_x\}_{x\in\{0,1\}^*}$ be an ensemble of distributions of the form $\mu_x : \{0,1\}^{q(|x|)} \to \mathbb{R}$ where q is a strictly increasing polynomial. Then,*

1. *$\{\mu_x\}_{x\in\{0,1\}^*} \in \textit{PComp}_{\text{STR}}$ if $\{\mu_x\}_{x\in\{0,1\}^*}$ has a signed-digit representation $\mu_x(y) = \sum_{k=1}^{\infty} a_{x,y,k} \cdot 2^{-k}$, $a_{x,y,k} \in \{-1,0,1\}$, and there exists a polynomial-time Turing machine M such that $M(x,y,0^k) = a_{x,y,k}$*

2. *$\{\mu_x\}_{x\in\{0,1\}^*} \in \textit{ExactPSamp}_{\text{STR}}$ if there exists a polynomial p and a probabilistic Turing machine M such that*

$$\forall x \in \{0,1\}^*,\ \forall y \in \{0,1\}^{q(|x|)},\ \left|\hat{\mu}_x(y) - \Pr\left(M_{p(|x|,i)}(x) = y\right)\right| \leq 2^{-i}$$

3. *$\{\mu_x\}_{x\in\{0,1\}^*} \in \textit{PSamp}_{\text{STR}}$ if there exists a polynomial p and a probabilistic Turing machine M*

such that

$$\forall x \in \{0,1\}^*,\ \forall y \in \{0,1\}^{q(|x|)},\ \left|\hat{\mu}_x(y) - \Pr\left(M_{p(|x|,i)}(x,0^i) = y\right)\right| \leq 2^{-i}$$

Unfortunately, $\mathit{ExactPSamp}_{\mathrm{STR}}$ and even $\mathit{PSamp}_{\mathrm{STR}}$ do not easily relate in the way one might hope to any of their previously discussed analogues. Consider for instance the ensemble $\{\mu_x\}_{x\in\{0,1\}^*}$ where μ_x denotes a distribution over $\{0,1\}^{|x|}$. This may be tranformed to a numerically indexed ensemble by considering x to have be drawn uniformly from $\{0,1\}^n$ and y from μ_x. Supposing that the pairing function satisfies $|\langle x,y\rangle| = c\cdot|x|$ for some constant c, then the ensemble $\{\pi_n\}_{n\in\mathbb{N}}$ consists of distributions $\pi_n : \{0,1\}^n \to \mathbb{R}$ defined by

$$\hat{\pi}_{c\cdot m}(\langle x,y\rangle) = \sum_{|x|=m} 2^{-n}\cdot\hat{\mu}_x(y)$$

As a formality, when n is not a multiple of c, μ_n is taken to be the uniform distribution over $\{0,1\}^n$.

Now it is not difficult to see that $\{\mu_x\}_{x\in\{0,1\}^*} \in \mathit{PSamp}_{\mathrm{STR}}$ implies $\{\pi_n\}_{n\in\mathbb{N}} \in \mathit{PSamp}_{\mathrm{NUM}}$ since a sample from $\pi_{c\cdot m}$ may be obtained simply by first sampling $x \in \{0,1\}^m$ with probability 2^{-n}, then sampling y from μ_x, and finally returning $\langle x,y\rangle$.

On the other hand, it is not evident that $\{\pi_n\}_{n\in\mathbb{N}} \in \mathit{PSamp}_{\mathrm{NUM}}$ implies $\{\mu_x\}_{x\in\{0,1\}^*} \in \mathit{PSamp}_{\mathrm{STR}}$. Given a sampling machine for $\{\pi_n\}_{n\in\mathbb{N}}$, the natural approach to producing samples from μ_x may be to sample from $\pi_{c\cdot|x|}$ until a sample of the form $\langle x,y\rangle$ is observed, at which point y is returned. However, since the probability of observing x is $2^{-|x|}$, the expected number of steps required will be $2^{\Omega(|x|)}$.

Chapter 8

Uniform Sampling of NP-witnesses

8.1 Jerrum, Valiant and Vazirani

So far we have only considered classes for which in some sense efficient sampling is possible. It is also useful to consider broader classes of distributions from which we might wish to produce samples efficiently. In this direction, inspiration may be taken from the dichotomy between P and NP. Jerrum, Valiant and Vazirani consider the task of sampling uniformly from the set of certificates to an accept instance of an NP-decision problem [7]. The task may be viewed as a way to observe a typical solution to the problem instance. In particular, Jerrum et al. consider both exact and approximate notions of uniform sampling which we introduce now.

A relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$ is called an *p-relation* if it is computable in deterministic polynomial time and is satisfied only by pairs of the form (x, y) where $|y| \leq p(|x|)$ for some fixed polynomial p. Given $\mathcal{L} \subseteq \{0,1\}^*$, then by definition $\mathcal{L} \in NP$ iff there exists a p-relation R such that

$$\forall x \, (x \in \mathcal{L} \Leftrightarrow \exists y \; R(x, y) = 1)$$

When $x \in \mathcal{L}$, the string y here is commonly referred to as a *certificate* of x.

Now given a p-relation R, witness to some $\mathcal{L} \in NP$, the exact sampling task is to provide a polynomial-time probabilistic machine M which on input x produces any one of the certificates y satisfying $R(x, y)$ with equal probability. Strings that are not certificates are not produced, so no output is produced if $x \notin \mathcal{L}$. Furthermore, when $x \in \mathcal{L}$, it is allowed that the machine does not produce a sample with probability at most $\frac{1}{2}$, an arbitrary constant which allows us to guarantee that samples are produced with high probability when the procedure is repeated iteratively. Formally, an *exact uniform generator* for R is a polynomial-time probabilistic machine M for which:

1. There exists a function $\psi : \{0,1\}^* \to (0,1]$ such that, for all $x, y \in \{0,1\}^*$,

$$\Pr(M(x) = y) = \begin{cases} \psi(x) & \text{if } \langle x, y \rangle \in R \\ 0 & \text{if } \langle x, y \rangle \notin R \end{cases}$$

2. For any input $x \in \{0,1\}^*$ where $\{y \in \{0,1\}^* : R(x,y)\}$ is non-empty,

$$\Pr(M(x) \neq \perp) \geq \frac{1}{2}$$

Supposing that a uniform generator M exists for R, it may be seen to represent the ensemble of distributions $\{\mu_x\}_{x\in\{0,1\}^*}$ where:

1. For $x \in \mathcal{L}$, μ_x is the distribution that assigns the same probability to all strings $y \in \{0,1\}^{p(x)}$ satisfying $M(x,y) = 1$ and null probability to everything else;

2. For $x \notin \mathcal{L}$, μ_x is the distribution which assigns probability 1 to $\perp$, and null probability to everything else.

Indeed, if M exists, then $\{\mu_x\}_{x\in\{0,1\}^*} \in \mathit{ExactPSamp}_{\mathrm{STR}}$. To see this, take $\overline{M}$ to be the machine which, on input $x \in \{0,1\}^*$, iteratively executes M on input x until an output $y \in \{0,1\}^*$ is produced, at which point $\overline{M}$ returns y. Then, conditioned on $\overline{M}$ having halted, the probability that $\overline{M}$ produces a particular string $y \in \{0,1\}^*$ is exactly $\mu_x(y)$. Furthermore, on each execution of M, the probability that M produces output is $\frac{1}{2}$, so that, after k iterations of M, the probability that $\overline{M}$ has not halted is 2^{-k-1}. Since k iterations of M require time $k \cdot t(|x|)$, where $t(|x|)$ is the running time of $\overline{M}$ on x, it follows that M approximates μ according to the following expression.

$$(1 - 2^{-k-1}) \cdot \mu_x(y) \leq \Pr(\overline{M}(x)_{k\cdot t(|x|)} = y) \leq \mu_x(y)$$

Thus $\overline{M}$ is a witness to $\{\mu_x\}_{x\in\{0,1\}^*} \in \mathit{ExactPSamp}_{\mathrm{STR}}$. Actually, the approximation is considerably stronger than needed since the error attained is multiplicative whereas $\mathit{ExactPSamp}_{\mathrm{STR}}$ requires only additive error.

It is primarily this context, namely the sampling of *NP*-witnesses, which causes us to allow distributions that assign positive probability to $\perp$. In particular, when $x \notin \mathcal{L}$, then $\mu_x(\perp) = 1$ and, hence, for all $y \in \{0,1\}^*$, $\mu_x(y) = 0$. Thus, the definition of $\mathit{ExactPSamp}_{\mathrm{STR}}$ allows the sampling machine $\overline{M}$ to not halt on input x. This is essential since an exact uniform generator M for $\mathcal{L}$ provides no mechanism for recognizing with certainty, after any amount of time, when $x \notin \mathcal{L}$.

When uniform generation is not possible for the given p-relation R, witness to $\mathcal{L} \in NP$, it may still be possible to solve the approximate version of the task. Here, in addition to the input $x \in \{0,1\}^*$, the machine takes an accuracy term $k \in \mathbb{Z}_{k\geq 1}$, which determines by how much the probabilities, of producing the various certificates y satisfying $R(x,y)$, are allowed to vary. Formally, a probabilistic machine M is called an *almost uniform generator* for R if:

1. There exists a function $\phi : \{0,1\}^* \to (0,1]$ such that, for all $\langle x,k\rangle \in \{0,1\}^* \times \mathbb{Z}_{\geq 1}$,

$$\langle x,y\rangle \in R \Rightarrow (1+1/k)^{-1}\phi(x) \leq \Pr(M(x,0^k) = y) \leq (1+1/k)\phi(x)$$

$$\langle x,y\rangle \notin R \Rightarrow \Pr(M(x,0^k) = y) = 0$$

2. For any input $\langle x, k\rangle$ where $\{y \in \{0,1\}^* : R(x,y)\}$ is non-empty,

$$\Pr(M(x, 0^k) \neq \perp) \geq \frac{1}{2}$$

Whereas it was possible to relate exact uniform generation to *ExactPSamp*, this does not appear to be the case in relating almost uniform generation to *PSamp*. In particular, the definition of *PSamp* uses additive error whereas almost uniform generators are defined with multiplicative error, which one should expect to be much more difficult to attain. On the other hand, the definition of *PSamp* requires that the error decreases exponentially in polynomial time whereas the definition of an almost uniform generator requires only that the error decreases polynomially in polynomial time. Thus the definitions of *PSamp* and of an almost uniform generator are not immediately comparable.

Jerrum et al. find that, for all $R \in NP_{\text{rel}}$, there exists a machine equipped with an *NP*-oracle that is an almost uniform generator for R. They also find that, for all $R \in NP_{\text{rel}}$, there exists a machine equipped with a Σ_2^p-oracle that is an exact uniform generator for R. These results alone may have provided some circumstantial evidence that almost uniform generation could be harder than exact uniform generation, since it appeared that stronger machinery was used to obtain the latter. However, it was later shown by Bellare, Goldreich and Petrank that, for all $R \in NP_{rel}$, access to an *NP* oracle is sufficient for exact uniform generation [2].

8.2 Main uniform sampling definitions

Because the definitions of Jerrrum, Valiant and Vazirani are not immediately comparable to our definitions of $PSamp_{\text{STR}}$ and $ExactPSamp_{\text{STR}}$, we introduce our own analgous definitions.

Definition 28. *NP_{rel} is the set of all p-relations. That is, for $R \subseteq \{0,1\}^* \times \{0,1\}^*$, then $R \in NP_{\text{rel}}$ iff R is computable in deterministic polynomial time and, for some polynomial p, satisfied only by pairs of the form (x, y) where $|y| \leq p(|x|)$.*

Definition 29. *Let $R \in NP_{\text{rel}}$ so that $(x, y) \in R$ implies $|y| \leq p(|x|)$ for some fixed polynomial p. Assume that R is witness to $L \in NP$. Then the corresponding ensemble $\{\mu_{R,x}\}_{x\in\{0,1\}^*}$ consists of distributions of the form $\mu_{R,x} : \{0,1\}^{\leq p(|x|)} \cup \{\perp\} \to \mathbb{R}$ defined, for $x \in L$, by*

$$\hat{\mu}_{R,x}(y) = \begin{cases} |\{y' : R(x,y')\}|^{-1} & \text{if } \langle x, y\rangle \in R \\ 0 & \text{otherwise.} \end{cases}$$

and, for $x \notin L$, by

$$\hat{\mu}_{R,x}(y) = \begin{cases} 1 & \text{if } y = \perp \\ 0 & \text{if } y \neq \perp \end{cases}$$

Definition 30. *$PS \subseteq NP_{\text{rel}}$ is the set of all p-relations R for which $\{\mu_{R,x}\}_{x\in\{0,1\}^*} \in PSamp_{\text{STR}}$.*

Definition 31. *$PS^* \subseteq NP_{\text{rel}}$ is the set of all p-relations R for which $\{\mu_{R,x}\}_{x\in\{0,1\}^*} \in ExactPSamp$.*

8.3 A notion of reduction

To demonstrate the existence of a complete sampling problems, a simple type of reduction is introduced, where the problem instance of one class is taken to a problem instance of the second class with an equal number of certificates. In turn, certificates of the second class are mapped to back to certificates of the first class by an efficiently computable bijection.

Definition 32. *Let $R, S \in NP_{\text{rel}}$. Then say that R is sample-to-sample reducible to S if there exist functions $f : \{0,1\}^* \to \{0,1\}^*$ and $g : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ such that*

1. $f, g \in FP$
2. $|\{y : R(x,y)\}| = |\{y : S(f(x),y)\}|$
3. *The function g_x defined by $g_x(y) = g(x,y)$ is a bijection from $\{y : S(f(x),y)\}$ to $\{y : R(x,y)\}$*

The following fact guarantees that both PS and PS^* are closed under these reductions.

Proposition 33. *Let $R, S \in NP_{\text{rel}}$ and suppose that R is sample-to-sample reducible to S. Then,*

1. $S \in PS^* \Rightarrow R \in PS^*$
2. $S \in PS \Rightarrow R \in PS$

Proof. Suppose that R is sample-to-sample reducible to S by way of the functions f and g as in Definition 32. Then consider the case where $S \in PSamp_{\text{STR}}$, as witnessed by the machine M. Take $\overline{M}$ to be the machine which, on input $(x, 0^i)$, computes $M(f(x), 0^i)$. Supposing that $M(f(x), 0^i) \neq \perp$, have $\overline{M}$ return $g(M(f(x), 0^i))$. Otherwise, $\overline{M}$ does not halt. It is easy to verify that $\overline{M}$ defined in this way is a witness to $\{\mu_{R,x}\}_{x \in \{0,1\}^*} \in PSamp_{\text{STR}}$. Therefore, $R \in PS$. A similar argument shows that $S \in PS^*$ implies $R \in PS^*$. ∎

Many p-relations may be shown to be complete for the class NP_{rel}. The p-relation SAT_{rel} is given for illustration.

Definition 34. *Let $SAT_{rel} \in NP_{\text{rel}}$ be the relation for which $(x, y) \in R$ iff x is the encoding of a boolean formula and y is the encoding of a satisfying assignment to x.*

The completeness of SAT_{rel}, and the completeness of a number of other p-relations, may be viewed as slight variations on the completeness of various counting problems, demonstrated by Leslie Valiant [11].

Proposition 35. *For all $R \in NP_{\text{rel}}$, it holds that R is sample-to-sample reducible to SAT_{rel}.*

Proof. First consider the special case where, for some polynomial $p : \mathbb{N} \to \mathbb{N}$, it holds that $R(x, y)$ is satisfied only by pairs of the form (x, y) where $|y| = p(|x|)$. For an arbitrary p-relation of this form, the standard proof that SAT is NP-complete constructs a polynomial-time computable mapping $x \mapsto \varphi_x$ from strings to boolean formulas such that $R(x, y)$ iff $\varphi_x(y \circ z)$ for some string z [4] [1]. Careful consideration of the construction involved leads to the stronger conclusion that, if $R(x, y)$,

then there exists a unique string z such that $\varphi_x(y \circ z)$. Assume that the mapping $x \mapsto \varphi_x$ has this property. Then let $f \in FP$ be the function defined by $f(x) = \varphi_x$. Also, let $g \in FP$ be the function which maps (x, v) to (x, y) where y is the length-$p(|x|)$ prefix of v. Since, for $y \in p(|x|)$, $\phi_x(y \circ z)$ implies $R(x, y)$, it follows that the function

$$g_x : \{y : SAT_{rel}(f(x), y)\} \to \{y : R(x, y)\}$$

given by $g_x(y) = g(x, y)$ is well-defined. Moreover, g_x is injective since strings in $\{y : S(f(x), y)\}$ are uniquely determined by their length-$p(|x|)$ prefix. That g_x is surjective follows from the fact that $R(x, y)$ implies $\phi_x(y \circ z)$ for some string z.

To apply this argument to the more general case where $R(x, y)$ is satisfied only by pairs of the form (x, y) where $|y| \leq p(|x|)$, it suffices to first convert R to a relation of the more restrictive form by way a padding scheme applied to the certificates y.

■

Corollary 36.

1. $SAT_{rel} \in PS \Rightarrow PS = NP_{\text{rel}}$
2. $SAT_{rel} \in PS^* \Rightarrow PS^* = NP_{\text{rel}}$

Proof. By Propositions 33 and 35. ■

8.4 Can completeness provide evidence for the separation of *PS* and *PS**?

While the existence of complete problems for NP_{rel} is useful in designating sampling tasks which one expects to be difficult or impossible to perform efficiently, it remains to be seen whether completeness may be leveraged in separating PS from PS^*.

Given a problem X in the class of computational problems B, it may be broadly defined, without reference to a particular formalization of reducation, that X is B-complete relative to the subclass $A \subseteq B$ if

$$X \in A \Rightarrow A = B$$

In this sense, Corollary 36 says that SAT_{rel} is both NP_{rel}-complete relative to PS and NP_{rel}-complete relative to PS^*.

If we were to find a problem $Q \in NP_{\text{rel}}$ that was shown to be both NP_{rel}-complete relative to PS^* and, at the same time, approximately samplable in the sense $Q \in PS$, this could provide strong evidence for the separation of PS and PS^*. In particular, it would require only the assumption that not all relations $R \in NP_{\text{rel}}$ have exact uniform generators, in other words $PS^* \neq NP_{\text{rel}}$, to obtain $PS \neq PS^*$.

However, if we are to make the slightly stronger but also reasonable assumption that $PS \neq NP_{\text{rel}}$, which is to say that not all relations $R \in NP_{\text{rel}}$ have almost uniform generators, then SAT_{rel} is not

a valid candidate for Q, since $SAT_{\text{rel}} \in PS$ implies $PS = NP_{\text{rel}}$. Likewise, it would follow that any relation $P \in NP_{\text{rel}}$, which is NP_{rel}-complete relative to PS, could not be a candidate for Q since then it would again follow from $Q \in PS$ that $PS = NP_{\text{rel}}$.

In short, in searching for a relation $Q \in NP_{\text{rel}}$ that is both NP_{rel}-complete relative to PS as well as a member of PS, we should expect Q to not be NP_{rel}-complete relative to PS^*. This motivates us to consider whether there is a useful form of reduction under which PS^* is closed but under which PS is not. It is precisely at this point that the paradigm begins to appear untenable since in general it is to be expected that a reduction which preserves samplability in a $ExactPSamp_{\text{STR}}$ sense should preserve samplability in the more general $PSamp_{\text{STR}}$ sense. It remains possible that the membership of a relation in PS^* may be exploited by some form of reduction so as to produce new relations in PS^*, in such a way that the same form of reduction cannot exploit membership in PS to produce even members of PS. Nevertheless, such a technique would be far removed from usual approaches to reduction in the context of computation. Lacking further direction on how such a barrier might be overcome, it is necessary to look elsewhere for strategies in proving the separation of $PSamp_{\text{STR}}$ and $ExactPSamp_{\text{STR}}$.

Chapter 9

Conclusion

This work has introduced the class *ExactPSamp* and considered the question of the separation of *PSamp* from previously studied sampling classes, most notably *ExactPSamp*. By consequence of the nearness of *ExactPSamp* and *PSamp* in terms of p-domination, we have noted that average-case complexity is agnostic with regards to these. In considering the task of transforming a machine that witnesses a distribution's membership in *PSamp* into a machine which witnesses the distribution's membership in *ExactPSamp*, we have noted basic barriers such as the fact that *PSamp* allows samples assigned null probability to be produced with small positive probability whereas this is disallowed by *PSamp*. Further considering challenges to the demonstration of $PSamp = ExactPSamp$, we have studied Yamakami's notion of monotonic sampling, formalized by the class *MonoPSamp*. We have extended these ideas of monotonicity to that of consistent monotonicity and defined the class *ConsistentPSamp* which, as it turns out, is equal to *ExactPSamp* and thereby provides an alternative characterization of the latter class. Characterization of each of *PSamp*, *MonoPSamp*, and *ExactPSamp* in terms of their random bittape configurations has also been provided. In the context of the study of the ensemble versions of our sampling classes, a practical advantage of *PSamp* has been demonstrated, namely that *PSamp* relates naturally to the analogous class of ensembles $PSamp_{\mathrm{NUM}}$, where *ExactPSamp* is not known to relate to $ExactPSamp_{\mathrm{NUM}}$ in the same way. Whereas $PSamp_{\mathrm{NUM}}$ and $ExactPSamp_{\mathrm{NUM}}$ consist of ensembles where the distributions are indexed over sample length, we have also considered the classes $PSamp_{\mathrm{STR}}$ and $ExactPSamp_{\mathrm{STR}}$ consisting of ensembles with their distributions indexed by strings, where the sample size is polynomial in the size of the index. This enables the discussion of sampling from the witnesses of an *NP* relation. In this context, we have considered using completeness to provide evidence for the separation of $PSamp_{\mathrm{STR}}$ and $ExactPSamp_{\mathrm{STR}}$, and it has been argued why this approach is likely to be fruitless.

Altogether, the question of the separation or collapse of *PSamp* and *ExactPSamp* has remained elusive. While it is not surprising that $PSamp \neq ExactPSamp$ has not been shown, since this would imply $P \neq PP$, it remains an open problem as to whether $PSamp \neq ExactPSamp$ may be obtained, even as a consequence of traditional complexity theoretic assumptions. Indeed, it may be overly optimistic to take for granted that this more modest goal may be attained. After all, it is due to the

phenomenon that the relationship between complexity classes is so often only weakly understood, that an abundant 'zoo' of complexity classes continues to be studied. Considering the subtlety of the difference between the definitions of *PSamp* and *ExactPSamp*, it is credible that the gap between *PSamp* and *ExactPSamp*, supposing it exists, might be in some sense small, relative to the gap between more traditional complexity classes, such as that which may exist between P and NP.

Bibliography

[1] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, New York, NY, USA, 1st edition, 2009.

[2] Mihir Bellare, Oded Goldreich, and Erez Petrank. Uniform generation of NP-witnesses using an NP-oracle. *Information and Computation*, 163(2):510–526, 2000.

[3] Shai Ben-David, Benny Chor, Oded Goldreich, and Michel Luby. On the theory of average case complexity. *Journal of Computer and System Sciences*, 44(2):193–219, 1992.

[4] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, pages 151–158, New York, NY, USA, 1971. ACM.

[5] Yuri Gurevich. Average case complexity. In *Automata, languages and programming (Madrid, 1991)*, volume 510 of *Lecture Notes in Comput. Sci.*, pages 615–628. Springer, Berlin, 1991.

[6] Russell Impagliazzo and Leonid A. Levin. No better ways to generate hard NP instances than picking uniformly at random. In *IEEE Annual Symposium on Foundations of Computer Science*, pages 812–821 vol. 2. 1990.

[7] Mark R. Jerrum, Leslie G. Valiant, and Vijay V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoret. Comput. Sci.*, 43(2-3):169–188, 1986.

[8] Leonid A Levin. Average case complete problems. *SIAM J. Comput.*, 15(1):285–286, February 1986.

[9] Peter Bro Miltersen. The complexity of malign measures. *SIAM J. Comput.*, 22(1):146–156, 1993.

[10] Luca Trevisan. Pseudorandomness in computer science and in additive combinatorics. In *An irregular mind*, volume 21 of *Bolyai Soc. Math. Stud.*, pages 619–650. János Bolyai Math. Soc., Budapest, 2010.

[11] Leslie G. Valiant. The complexity of enumeration and reliability problems. *SIAM J. Comput.*, 8(3):410–421, 1979.

[12] Tomoyuki Yamakami. Polynomial time samplable distributions. *J. Complexity*, 15(4):557–574, 1999.