How to sample from $\mathbb{N}$ and from $\Sigma^\star$, and why*
(extended abstract)

Stavros Konstantinidis
Mathematics and Computing Science,
Saint Mary's University, Halifax, NS, Canada
s.konstantinidis@smu.ca

This exposition is based on past and ongoing research with Pantelis Andreou, Mitja Mastnak, Nelma Moreira, Argyrios Papageorgiou, Rogério Reis, Juraj Šebej, and Taylor Smith. It was presented on November 4, 2026, at the EAADS Workshop in Košice, Slovakia.

1 Motivation

Consider a hard NFA/REGEXP problem like universality, or equivalence. We want a PTIME randomized test (algorithm) to give us information about the hard problem. Here is the typical test for a given instance α of the problem:

```
Testing algorithm A(α,ε)
    sample n = n(ε) words from Σ*
    if a witness is found
            return answer with 100% certainty
    else  # no witness
            return what?
```

For the case of NFA universality, we can say that if no witness word is found then the given NFA α is probably ε-close to being universal, in the sense that the set of words that are not accepted by the given NFA α *has a mass* $\leq \varepsilon$. This is meaningful in the theory of codes: We want to know whether a certain type of code is *close to being maximal*. It can be meaningful in Boolean expression satisfaction: We want to know whether a Boolean expression is *close to being a tautology*. Consider now "hard" mathematical problems about nonnegative integers. For example, does the Diophantine equation $x^3y^2 - z^3 - 6 = 0$ have a solution, [11]? Again, here is the test: sample a number n of nonnegative integer triples (x,y,z); if a witness is found the answer is 100% correct; else the set of witnesses has a small mass.

In order to talk about the mass/probability of a subset of $\mathbb{N}$, or $\Sigma^\star$, we need to define a probability distribution D on the elements of $\mathbb{N}$, or of $\Sigma^\star$, such that sampling from D can be done in PTIME. The set $\mathbb{N}$, or $\Sigma^\star$, is called the domain of the distribution. The key point here is that the domain is (countably) infinite. Similar hard questions can be defined if the sampling domain is finite. In this case, the standard approach is to use as D the *uniform distribution*.

The problems that we have recently investigated, or we are currently investigating, are the following.

Problem 1 There is no uniform distribution on $\mathbb{N}$, or on $\Sigma^\star$. What is a good substitute?

Problem 2 Once a distribution D is chosen, how to sample from D in PTIME? This requires limiting sampling to points/objects of some maximum value/size M.

Problem 3 How to determine the sample size n and the maximum value/size M of the points/objects being sampled?

*This work was funded by a Discovery Grant of NSERC Canada (grant number RGPIN-2020-05996).

Publication:
Technical Report 2026-002, Dept Math. and Computing Sci., Saint Mary's University, Halifax, NS, Canada

These problems are investigated within the framework of PRAX (Polynomial Randomized Approximation) algorithms [16, 17, 2]—see Appendix.

Problem 4 Which hard problems have a good PRAX algorithm?

2 Problem 1: determining good distributions on $\mathbb{N}$ and $\Sigma^\star$

We use the notation $\mathbb{N}$ for the set of positive integers, $\mathbb{N}_0$ for the nonnegative integers, and $\mathbb{N}_0^{>x}$ (resp., $\mathbb{N}_0^{\leq x}$) for the nonnegative integers greater than x (resp., less than or equal to x), where x is any real number.

Probability Distributions, [9, 10, 16]. Let X be a countable set. A probability distribution on X is a function $D : X \to [0,1]$ such that $\sum_{x\in X} D(x) = 1$. The domain of D, denoted by dom$D$, is the subset $\{x \in X : D(x) > 0\}$ of X. If X is *finite*, that is, $X = \{x_1, \ldots, x_k\}$ for some $k \in \mathbb{N}$, then we write

$$D = \big(D(x_1), \ldots, D(x_k)\big).$$

If $X \subseteq \mathbb{N}_0$ then the distribution D is called a length distribution, and if $X \subseteq \mathbb{N}_0 \times \mathbb{N}_0$ then the distribution D is called a 2D length distribution—see below the Dirichlet length distributions. We use the following terminology from [10].

Definition 2.1. *Let D be a probability distribution on X. For any subset L of X, we define the quantity*

$$D(L) = \sum_{x\in L} D(x) \tag{1}$$

and refer to it as the probability that a randomly selected element from D is in L, *or as* the D-mass of L.

Dirichlet distributions. For any $t > 1$ and $d \in \mathbb{N}_0$, the Dirichlet distribution $\mathsf{D}_{t,d}$ on $\mathbb{N}_0$ is defined such that

$$\mathsf{D}_{t,d}(\ell) = (1/\zeta(t))(\ell + 1 - d)^{-t}$$

for $\ell \in \mathbb{N}_0^{\geq d}$, [9, 16], where ζ is the Riemann zeta function. This is a length distribution. The original version in [9] uses no parameter d (i.e., $d = 1$), however here as in [16], we allow $d > 0$ when it is desirable to skip small lengths. In [9] the author considers the Dirichlet distribution to be the basis where *"many heuristic probability arguments based on the fictitious uniform distribution on the positive integers become rigorous statements."* For some applications, we need 2D versions of $\mathsf{D}_{t,d}$:

$$\mathsf{D}^2_{t,d}(k,\ell) = \mathsf{D}_{t,d}(k)\mathsf{D}_{t,d}(\ell),$$

which can be used to select two independent lengths. This is a 2D length distribution.

Word distributions. A length distribution D can be used as a first step to select a word in $\Sigma^\star$: first select a length ℓ with probability $D(\ell)$ and then select uniformly from Σ each of the ℓ symbols of the word. This process defines the word distribution $\langle D\rangle$ such that

$$\langle D\rangle(w) = D(|w|) \cdot s^{-|w|}, \quad \text{for all } w \in \Sigma^*.$$

We have that $\langle D\rangle(\Sigma^\ell) = D(\ell)$, for any word length ℓ.

The bias of $\mathsf{D}_{t,d}$. The Dirichlet distribution $\mathsf{D}_{t,d}$ is meant to be a heuristic for the fictitious uniform distribution on $\mathbb{N}_0^{\geq d} = \{d, d+1, d+2, \ldots\}$. We focus on the case of $d = 1$; hence, the sample space is $\mathbb{N} = \{1, 2, 3, \ldots\}$. As $t \to 1^+$, $\mathsf{D}_{t,d}$ becomes better, in the sense that certain basic desirable probabilities

on $\mathbb{N}$ are respected. For example, the probability that a number ℓ selected from $\mathsf{D}_{t,d}$ is a multiple of m, for given $m \in \mathbb{N}$, is

$$\mathsf{D}_{t,d}(\ell \% m = 0) = 1/m^t,$$

which gets closer to the desired mass $1/m$ of the subset of multiples of m, as $t \to 1^+$. In fact, we have that, [9], the probability that a number ℓ selected from $\mathsf{D}_{t,d}$ satisfies $\ell \% m = r$, for any fixed $r = 0, \ldots, m-1$, tends to $1/m$ as $t \to 1^+$. Thus, the mass of the set of even numbers is $1/2^t$, which is quite biased if t is not close enough to 1. A similar observation holds for the mass of the set of numbers that are multiples of three. This is because $\mathsf{D}_{t,d}$ assigns a relatively large probability to $\mathsf{D}_{t,d}(1)$. For example, for $t = 1.4$, we have $\mathsf{D}_{t,d}(1) = \frac{1}{\zeta(1.4)} \approx 0.322$. This means that $\mathsf{D}_{t,d}(1)$ alone gets more than **32%** of the total probability mass, which may be undesirable for some applications. We want to somehow mitigate this bias by taking some mass from a few initial values $\mathsf{D}_{t,d}(1), \mathsf{D}_{t,d}(2), \mathsf{D}_{t,d}(3), \ldots$ and distributing it to later values.

Note. The reader might wonder why we do not simply consider a distribution $\mathsf{D}_{t,d}$ with a value of t close to 1, e.g., $t = 1.1$. It turns out that this choice would entail an enormously large maximum value/size of the points/objects being sampled—see also Sections 3,4.

Next we present an ongoing attempt to define a variant $\mathsf{D}'_{t,d}$ of $\mathsf{D}_{t,d}$. We use $d = 1$, so that the sampling space is $\mathbb{N} = \{1, 2, 3, \ldots\}$.

Objectives. In defining the Dirichlet variant $\mathsf{D}'_{t,d}$, the objectives are: (i) for any $m \geq 2$, the $\mathsf{D}'_{t,d}$-mass of the multiples of m is closer to the desired value $1/m$ than the $\mathsf{D}_{t,d}$-mass (which is $1/m^t$) of those multiples; (ii) the mass (probability) $\mathsf{D}'_{t,d}(1)$ of the number 1 is smaller than $\mathsf{D}_{t,d}(1)$.

Initial idea. In $\mathsf{D}'_{t,d}$ we can reduce $\mathsf{D}_{t,d}(1)$ to some value $h < \mathsf{D}_{t,d}(1)$ and then redistribute the amount $\mathsf{D}_{t,d}(1) - h$ across the rest of the values $\mathsf{D}_{t,d}(\ell)$:

$$\mathsf{D}'_{t,d}(1) = h \quad \text{and} \quad \mathsf{D}'_{t,d}(\ell) = \mathsf{D}_{t,d}(\ell) \cdot \frac{1-h}{1-\mathsf{D}_{t,d}(1)} \text{ for } \ell \geq 2.$$

Revised idea. We extend the above idea as follows. For any integer $B \geq 1$, let $S_t(B)$ = the total mass of the first B values of $\mathsf{D}_{t,d}$, that is, $S_t = \sum_{\ell=d}^{d+B-1} \mathsf{D}_{t,d}(\ell)$—note here that S_t is independent of the starting index d. To define $\mathsf{D}'_{t,d}$, we need to pick a block length B and a value h (the head value) such that

1. the first B values of $\mathsf{D}'_{t,d}$ are all equal to h;
2. the total mass hB of the first B values is at most $S_t(B)$, that is, $hB \leq S_t(B)$;
3. the total mass of the rest of the values is $1 - hB$.
4. the $(B+1)$st probability value does not exceed the earlier values h: $\mathsf{D}'_{t,d}(d+B) \leq h$.

The 2nd and 3rd requirements take mass out of $S_t(B)$, the first B values, and redistribute it across the values $\ell \geq d+B$. As in [16], we also want to apply the principle of information theory assigning smaller probability values to larger sizes; that is where the 4th requirement comes in. So using B and h we define:

$$\mathsf{D}'_{t,d}(\ell) = \begin{cases} h, & \text{if } \ell = d, \ldots, d+B-1; \\ \rho \cdot \mathsf{D}_{t,d}(\ell), & \text{if } \ell \geq d+B; \end{cases} \qquad \text{where } \rho = \tfrac{1-hB}{1-S_t(B)}. \tag{2}$$

But how do we pick B and h? This is part of our ongoing research. Here is a selected result using a particular choice of a Dirichlet variant $\mathsf{D}'_{t,d}$ that addresses our Objective about the mass of the multiples of m. Moreover, the difference between $\mathsf{D}'_{t,d}(\ell \% m = 0)$ and $1/m$ tends to 0, as $m \to \infty$, and is always less than 0.0161.

Proposition 2.2. *Let $d = 1, t = 1.40, h = 0.02898, B = 9$. For all $m \geq 1$, we have that*

1. $|\mathsf{D}'_{t,d}(\ell\%m = 0) - 1/m| < 1/m - 1/m^t$,
2. $0.0161 > |\mathsf{D}'_{t,d}(\ell\%m = 0) - 1/m| \to 0$.

3 Problem 2: sampling from $\mathbb{N}$ and $\Sigma^\star$ in PTIME

Selecting from a distribution D with a large domain $X = \mathsf{dom}D$ could return an element of large size, which can be intractable with respect to the algorithmic size of the input. For example, selecting from $D = \mathsf{D}_{t,d}$ could return an arbitrarily large length, albeit with low probability. For this reason, given a desirable "small" positive $\varepsilon < 1$, we can choose a finite subset F of X and we select elements from F, omitting the (usually infinite) tail $X - F$ of D, provided that $D(X - F) \leq \varepsilon$. More specifically, if $F = \{x_1, \ldots, x_k\}$, the F-truncated version of D is the distribution D^F with domain $F \cup \{\perp\}$, where '$\perp$' is an object outside of X, such that

$$D^F = \big(D(x_1), \ldots, D(x_k),\, 1 - D(F)\big) \tag{3}$$

Each object $x \in X$ has an algorithmic size and we choose F such that the sizes of its objects are not too large w.r.t. the input size of the algorithm in which the selection takes place. Thus, the distribution D^F can select the outcome '$\perp$' instead of a large X-object.

A typical *algorithm* in which sampling is involved has the heading $A(\alpha, \varepsilon)$, where α is the object for which the algorithm is supposed to give an estimate (NFA, Boolean expression, Diophantine equation, etc), and ε is the approximation tolerance which is used to determine the number n of samples as well as a tail $X - F$ of D such that

- $D(X - F) \leq \varepsilon$, and
- the maximum size M of a sample object from D^F is $O\big((1/\varepsilon)^m\big)$ for some integer $m > 0$.

Example 3.1. The main example of a truncated length distribution D in [16] is the distribution D^F where $F = \mathbb{N}_0^{\leq M}$, for some desirable maximum length/size M. This distribution truncates all lengths $> M$. In fact in [16], D^F is denoted by D^M and is called an augmented distribution.

We assume a fixed distribution D for the algorithm $A(\alpha, \varepsilon)$. In fact, fixed is a family $D = (D_\alpha)$ of distributions, indexed by the possible instances α. For example, if the set of instances is the set of NFAs then each NFA α over some alphabet Σ specifies a word distribution D_α over the alphabet Σ, and D_α is usually a word distribution of the form $\langle \mathsf{D}_{t,d} \rangle$ or $\langle \mathsf{D}'_{t,d} \rangle$.

The algorithm $A(\alpha, \varepsilon)$ needs to precompute the finite distribution D^F in (3) before sampling can be performed. In fact it is more convenient (and equivalent) to precompute the cumulative distribution

$$\big(C(1),\, C(2), \ldots,\, C(k),\, 1\big), \tag{4}$$

where $C(1) = D(x_1)$, $C(2) = D(x_1) + D(x_2), \ldots, C(k) = D(F)$. Then, sampling from D^F can be done by first uniformly selecting a random[1] real $x \in (0, 1)$; then finding the first index i such that $x \leq C(i)$; and then returning, either the object x_i or $\perp$.

Note: *Following [3, pg 126] and [16], we assume unit cost for all basic arithmetic operations.*

[1] This is an ideal assumption. In practice, programming languages provide a function for selecting a pseudorandom rational number in $(0, 1)$.

4 Problem 3: the sample size n and the maximum sample value M

Consider again a typical *algorithm* $A(\alpha,\varepsilon)$ in which sampling is involved according to a certain distribution D, where α is the object for which the algorithm is supposed to give an estimate and ε is the approximation tolerance.

How to determine the sample size n. The sample size depends on what the sample is used for.

1. If the sample is needed for determining approximate universality, or emptiness, then the sample size n is linear, [2]:
$$n \geq \left\lceil \frac{1.4}{\varepsilon} \right\rceil \quad \text{or} \quad n \geq \left\lceil \frac{2.8}{\varepsilon} \right\rceil, \tag{5}$$
depending on whether the sampling domain is finite or infinite (in which case it will be truncated).

2. If the objective is to estimate a parameter p of the sampling domain, for example p = the probability that the number ℓ selected from $\mathsf{D}_{t,d}$ is even, then the sample size n is:
$$n \geq \left\lceil \frac{3\ln(2/\delta)}{\varepsilon^2} \right\rceil \quad \text{or} \quad n \geq \left\lceil \frac{12\ln(2/\delta)}{\varepsilon^2} \right\rceil, \tag{6}$$
depending on whether the sampling domain is finite, or infinite (in which case it will be truncated), [20, 2]. The above sample size (in either case) ensures that
$$\mathrm{P}[|p-\hat{p}| \geq \varepsilon] \leq \delta,$$
where $\delta \in (0,1)$. The value $1-\delta$ is the confidence in the sample estimate $\hat{p}$. In both cases, the parameter p to estimate is the mass of a subset $\mathscr{L}(\alpha)$ of the domain of D, that is, $p = D\big(\mathscr{L}(\alpha)\big)$.

How to estimate the maximum size M of a sampled object. Here the distribution D is either $\mathsf{D}_{t,d}$ or $\mathsf{D}'_{t,d}$. We have the following version of a lemma from [16, Lemma 6]. The lemma implies that we can truncate $\mathsf{D}_{t,d}$, which has an infinite domain, to $\mathsf{D}_{t,d}^{\mathbb{N}_0^{\leq M}}$, whose domain is finite such that the set of truncated values has probability at most ε, for any desirable tolerance ε. For simplicity, we write $\mathsf{D}_{t,d}^{M}$ for $\mathsf{D}_{t,d}^{\mathbb{N}_0^{\leq M}}$; hence,
$$\mathsf{D}_{t,d}^{M} \;=\; \big(\mathsf{D}_{t,d}(1),\ldots,\mathsf{D}_{t,d}(M),\, \mathsf{D}_{t,d}(\mathbb{N}_0^{>M})\big),$$
where of course $\mathsf{D}_{t,d}(\mathbb{N}_0^{>M}) = \sum_{i=M+1}^{\infty} \mathsf{D}_{t,d}(i)$.

Lemma 4.1. *Let $\varepsilon \in (0,1)$ and $M \in \mathbb{N}$. If $M \geq \sqrt[t-1]{1/\varepsilon} + (d-1)$ then $\mathsf{D}_{t,d}(\mathbb{N}_0^{>M}) \leq \varepsilon$.*

How to cut a tail of $\mathsf{D}'_{t,d}$ whose mass is $\leq \varepsilon$, for given approximation tolerance ε?

Lemma 4.2. *Let $\rho = \frac{1-hB}{1-S_t(B)}$, $\varepsilon \in (0,1)$ and $M = \lceil \sqrt[t-1]{\rho/\varepsilon} \rceil + (d-1)$. If $M \geq d+B-1$ then the tail $\mathsf{D}'_{t,d}(M+1), \mathsf{D}'_{t,d}(M+2), \ldots$ of $\mathsf{D}'_{t,d}$ has a total mass $\leq \varepsilon$. If $M < d+B-1$ then the tail $\mathsf{D}'_{t,d}(d+B), \mathsf{D}'_{t,d}(d+B+1), \ldots$ of $\mathsf{D}'_{t,d}$ has a total mass $\leq \varepsilon$. In the latter case, the non-truncated part is exactly the fist block of mass hB.*

Note. The above estimates for the value of M can be higher (not unreasonably higher) than the actual value of M that is computed as part of the precomputing process described in the previous section.

5 Some Experiments

Example 5.1. We want to experiment with some h values that would entail a mass $\mathsf{D}'_{t,d}(\ell\%m=0)$ closer to $1/m$ than the mass $\mathsf{D}_{t,d}(\ell\%m=0)=1/m^t$ is to $1/m$. In the following tests, we see that the mass of the multiples of m relative to $\mathsf{D}'_{t,d}$ is closer to the desired value $1/m$ than the mass $1/m^t$ relative to the original Dirichlet distribution $\mathsf{D}_{t,d}$.

```
printVarDirichletVals(15, 1, 1.40, 0.02898, 9)
Test for d=1, t=1.40, h=0.02898 and B=9
Variant Dirichl  D'-mass of multiples   D'-Mass - 1/m    1/m^t - 1/m
D'(002)=0.02898, D'(ell%002=0)=0.49326, Diff'=-0.00674,  Diff=-0.12107
D'(003)=0.02898, D'(ell%003=0)=0.32328, Diff'=-0.01005,  Diff=-0.11854
D'(004)=0.02898, D'(ell%004=0)=0.23840, Diff'=-0.01160,  Diff=-0.10641
D'(005)=0.02898, D'(ell%005=0)=0.18998, Diff'=-0.01002,  Diff=-0.09494
D'(006)=0.02898, D'(ell%006=0)=0.15371, Diff'=-0.01296,  Diff=-0.08527
D'(007)=0.02898, D'(ell%007=0)=0.12950, Diff'=-0.01336,  Diff=-0.07726
D'(008)=0.02898, D'(ell%008=0)=0.11236, Diff'=-0.01264,  Diff=-0.07059
D'(009)=0.02898, D'(ell%009=0)=0.09968, Diff'=-0.01143,  Diff=-0.06497
D'(010)=0.02897, D'(ell%010=0)=0.08998, Diff'=-0.01002,  Diff=-0.06019
D'(011)=0.02535, D'(ell%011=0)=0.07874, Diff'=-0.01217,  Diff=-0.05607
D'(012)=0.02245, D'(ell%012=0)=0.06971, Diff'=-0.01362,  Diff=-0.05249
D'(013)=0.02007, D'(ell%013=0)=0.06232, Diff'=-0.01460,  Diff=-0.04935
D'(014)=0.01809, D'(ell%014=0)=0.05618, Diff'=-0.01525,  Diff=-0.04657
D'(015)=0.01642, D'(ell%015=0)=0.05101, Diff'=-0.01566,  Diff=-0.04410
```

Example 5.2. The below table shows, for each $t \in \{1.50, 1.40, 1.35, 1.30, 1.25, 1.20\}$ and for each $\varepsilon \in \{0.05, 0.01, 0.005\}$: the maximum size M of a sampled object for the Dirichlet distribution $\mathsf{D}_{t,d}$ according to Lemma 4.1, and the maximum size M' of a sampled object for the variant Dirichlet distribution $\mathsf{D}'_{t,d}$ according to Lemma 4.2. For example, for $t=1.35, \varepsilon=0.01$, we have $M=517,948$ and $M'=3,752,939$. The two float numbers between M, M' exhibit the bias of $\mathsf{D}_{t,d}$: the mass of 1 is 28.9% and the mass of the even numbers is 39.2%.

t\eps	0.050	0.010	0.005
1.50	400	10,000	40,000
	0.383,0.354	0.383,0.354	0.383,0.354
	1,600	40,000	160,000
1.40	1,789	100,001	565,686
	0.322,0.379	0.322,0.379	0.322,0.379
	10,120	565,686	3,200,001
1.35	5,215	517,948	3,752,939
	0.289,0.392	0.289,0.392	0.289,0.392
	37,785	3,752,939	27,193,000
1.30	21,716	4,641,589	46,784,284
	0.254,0.406	0.254,0.406	0.254,0.406

```
                  218,877             46,784,284            471,556,032
-------------------------------------------------------------------------
1.25              160,000            100,000,000          1,600,000,000
                0.218,0.420           0.218,0.420            0.218,0.420
                  2,560,000         1,600,000,000         25,600,000,000
-------------------------------------------------------------------------
1.20              3,200,001         10,000,000,001        320,000,000,001
                0.179,0.435           0.179,0.435            0.179,0.435
                102,400,001       320,000,000,001       10,240,000,000,001
```

The maximum size M' in $\mathsf{D}'_{t,d}$ can be much larger than the maximum size M in $\mathsf{D}_{t,d}$ for the *same value of* t. However, one can simply use a higher t value with $\mathsf{D}'_{t,d}$ than with $\mathsf{D}_{t,d}$, which would entail a milder value for M'. This is demonstrated below.

A test for the mass of co-prime integers.
The below `ipython` script uses the Dirichlet variant $\mathsf{D}'_{t,d}$ to provide an estimate of the mass of the set of integer pairs (x,y) with $x,y \in \mathbb{N}$ that are co-prime, which is, [12, 19]:

$$6/\pi^2 \approx 0.607927101854026$$

For $\varepsilon = 0.01$, the sample size $n = 635,799$ according to the second bound in (6) is rather large. In fact, in the case of $t = 1.4$, that value exceeds the maximum sample value $M = 446,559$. It turns out that the bound is much larger in practice than necessary: for $\varepsilon = 0.01$, a sample size of $n \leq 50,000$ suffices.

The below tests generate two sets of samples (x,y) with $x,y \in \mathbb{N}$:

one for $t = 1.4, h = 0.02898, B = 9$ and one for $t = 1.35, h = 0.03771, B = 6$.

In both cases, the estimates for $6/\pi^2$ are reasonably good. In particular, for the first case of $t = 1.4$, the estimate is appropriate even for error tolerance $\varepsilon = 0.005$.

```
### Using f_VarDirichlet(n, d=1, t=1.4, head=0.02898, block=9):
G = GenSizeDis(f_VarDirichlet, err=0.01, times=5, delta=0.01)
### above took between 1 and 2 mins
prax_parameters(G)
(1400, 635799, 446559)
test_CoPrimePairsDis(G, 635799)
0.6138244948482147
test_CoPrimePairsDis(G, 50000)
0.6124

### Using f_VarDirichlet(n, d=1, t=1.35, head=0.03771, block=6):
G135 = GenSizeDis(f_VarDirichlet, err=0.01, times=5, delta=0.01)
### above took about 6 mins
prax_parameters(G135)
(1400, 635799, 1619849)
test_CoPrimePairsDis(G135, 635799)
0.6187269876171557
test_CoPrimePairsDis(G135, 50000)
0.61916
```

Below we run essentially the same code (case of $t = 1.4$) for varying sample size values

$$n \in \{100, 200, 500, 1000, 2000, 5000, 10000, 20000, 50000, 100000\}.$$

For each n, we take 500 estimates and compute the mean, standard deviation, and variance.

```
n = 100
Mean:  0.61172
Stdev: 0.0467655
Variance: 0.0021870
```

```
n = 200
Mean:  0.61095
Stdev: 0.0406638
Variance: 0.0016535
```

```
n = 500
Mean:  0.61122
Stdev: 0.0353064
Variance: 0.0012465
```

```
n = 1000
Mean:  0.61132
Stdev: 0.0316365
Variance: 0.0010008
```

```
n = 2000
Mean:  0.61167
Stdev: 0.0287026
Variance: 0.0008238
```

```
n = 5000
Mean:  0.61175
Stdev: 0.0263590
Variance: 0.0006948
```

```
n = 10000
Mean:  0.61181
Stdev: 0.0244726
Variance: 0.0005989
```

```
n = 20000
Mean:  0.61187
Stdev: 0.0229273
Variance: 0.0005256
```

```
n = 50000
Mean:  0.61194
Stdev: 0.0216293
Variance: 0.0004678
```

```
n = 100000
Mean:  0.61200
Stdev: 0.0205258
Variance: 0.0004213
```

Note. If we use the original Dirichlet distribution $\mathsf{D}_{t,d}$ for $t = 1.3$, the estimate of the mass of co-prime positive integers is greater than 0.75, which is not satisfactory.

Ongoing and future research. The above tests are part of our ongoing research. We plan to produce sample sets for future use in order to confirm theoretical results, or make predictions about unknown parameters, or for use in PRAX algorithms on standard automata, or 2D automata [7, 22]. For example: (i) for regular expressions α with intersection, as $|\alpha| \to \infty$ and the alphabet size $\to \infty$, we should have that, [5], the ratio $|\alpha|/|\alpha|_\cap$ should be 6; (ii) for regular expressions α in strong star normal form, we have that, [15], the average number of concatenations in the different partial derivatives of α by any single symbol is at most 14, as $|\alpha| \to \infty$ and the alphabet size $\to \infty$.

Software. Our experiments use FAdo, [1, 21], as a backbone library.

6 Problem 4: Which Problems have a good PRAX Algorithm?

PRAX algorithms (see Appendix) were introduced in [16] with the aim of providing a polynomial test for approximate NFA universality. As discussed in [16], neither approximation by itself, nor randomization by itself, is sufficient for the desired test:

Theorem 6.1. *The following problem about block NFAs*[2] *over* $\{0,1\}$ *is coNP-hard*

$$B_\delta = \{\boldsymbol{a} \in \mathsf{BNFA} : \frac{|\mathscr{L}(\boldsymbol{a})|}{2^\ell} \geq \delta, \text{ where } \ell = \text{word length of } \boldsymbol{a}\},$$

for any (fixed) $\delta \in (0,1)$ *that is computable within polynomial time*[3].

[2] A block NFA is an NFA $\boldsymbol{b}$ accepting only words of some fixed length ℓ. The NFA $\boldsymbol{b}$ is block-universal if it accepts Σ^ℓ.

[3] A real $x \in (0,1)$ is computable if there is an algorithm that takes as input a positive integer n and computes the n-th bit of x. It is computable within polynomial time if the algorithm works in time $O(n^k)$, for some fixed $k \in \mathbb{N}_0$, when the input n is given in unary.

Moreover, as NFA universality, or even block NFA universality, is hard, it is unlikely that the problem is in the standard randomized classes **RP** or co**RP**.

In [16, 17], the authors show that there is a PRAX algorithm for NFA universality and for NFA equivalence. In [2], the authors set the stage for the PRAX treatment of universality and emptiness problems in various domains:

Theorem 6.2. *Let (T_α) be a polynomially samplable, or a tractable, family of distributions (with some associated subset operator $\mathscr{L}$) such that it is polynomially decidable, for given α and $x \in \mathsf{dom}T_\alpha$, whether $x \in \mathscr{L}(\alpha)$. Then, there is a PRAX algorithm for the universality problem $U_T = \{\alpha \mid T(\alpha) = 1\}$; hence, also for the non-universality problem $\bar{U}_T$ as well as the emptiness and non-emptiness problems $E_T, \bar{E}_T$.*

In [2], the above result has been applied to obtain PRAX algorithms for emptiness of 2D-automata, and emptiness of solutions of certain Diophantine equations.

Ongoing and future research. In our work in progress we have obtained the following result.

Theorem 6.3. *Every* **NP** *problem has a PRAX algorithm relative to a family of finite* uniform *distributions.*

Remark 6.4. The above result says that in principle it is possible to define a polynomially samplable uniform distribution that can be used by a PRAX algorithm to give an approximate answer to any **NP** problem of interest. This result is mostly of theoretical interest: for each polynomial nondeterministic Turing machine T and for each instance α of T, the proof defines a finite sampling domain X_α that consists of all computations of T on α as well as a set of useless objects such the domain X_α is polynomially samplable and satisfies Theorem 6.2. The set of useless objects could be so large that the sampling process will need a very small ε in order to discover any successful computation of T on α, and this, although of polynomial cost, could be prohibitive in practice. On the other hand, there are hard problems for which the domain can be sampled very well in practice.

What about PSPACE problems? A stated above, there are PRAX algorithms for NFA universality and NFA equivalence. These algorithms use sampling from *truncated* infinite distributions. We believe that, for **PSPACE**-complete problems, it is not possible to have PRAX algorithms that use sampling from a (non-truncated) uniform distribution as in Theorem 6.3.

Performance of sampling on different problems. How do PRAX algorithms perform on different problems? How often do they fail? A PRAX universality (resp. emptiness) algorithm $A(\alpha, \varepsilon)$ fails exactly when there is no witness found and the instance α is not ε-close to being universal (resp. not ε-close to being empty). For the application of PRAX algorithms in testing maximality of variable length codes, we found in our experiments that it is not easy to fail the algorithm [16]. We also note that several PSPACE-complete problems remain hard even when the domain (set of instances) of the problem is restricted. For example, for NFA universality, the problem remains hard when the given NFA is partially ordered [18]. Also, for DFA intersection nonemptiness, the problem remains hard when the given DFAs accept languages low in the dot-depth hierarchy (or Straubing-Thérien hierarchy) [4]. A question that arises here is whether PRAX algorithms would work better, or can be tuned to work better, on restricted versions of hard problems.

Universality of NFAs resulting from operations with high state complexity. Many operations on DFAs produce DFAs of super-polynomial state complexity (in the worst case), while the nondeterministic state complexity of the operation is polynomial. For example, [6], given a DFA $\boldsymbol{a}$, the state complexity of the square operation $\mathscr{L}(\boldsymbol{a})^2$ is exponential, but the nondeterministic state complexity is polynomial. For

the universality question of whether $\mathscr{L}(\boldsymbol{a})^2 = \Sigma^\star$ one can apply a PRAX algorithm to get a randomized approximate answer. Note that the state complexity of the said operation remains exponential even if the given DFA $\boldsymbol{a}$ is ordered (see [13]). On the other hand, a PRAX algorithm is not necessary when universality can be decided efficiently with a non-randomized test, or even trivially; for example, while the state complexity of the boundary operation $\mathscr{L}(\boldsymbol{a})^* \cap \big(\overline{\mathscr{L}(\boldsymbol{a})}\big)^*$ is exponential, [14], the resulting language is never equal to $\Sigma^\star$.

Appendix: PRAX Algorithms

Following [2], we consider a set of instances α such that each instance is a subset description, that is, an expression (word) α that specifies a domain (a set) X_α and a *subset* $\mathscr{L}(\alpha)$ of X_α. For example, α could be (the representation of) an NFA with $X_\alpha = \Sigma^*$, for some alphabet Σ, and $\mathscr{L}(\alpha)$ = the language accepted by α; or α could be a Boolean expression, having some number m of variables, with $X_\alpha = \{\mathsf{True}, \mathsf{False}\}^m$ = all possible truth assignments to the m variables, and $\mathscr{L}(\alpha)$ = the truth assignments that satisfy α. We assume that membership of a given word $w \in X_\alpha$ in $\mathscr{L}(\alpha)$, for given α, is polynomially decidable. On the other hand, the universality problem asks whether the subset $\mathscr{L}(\alpha)$ is equal to X_α, and this problem is hard in many domains. For this reason, we want to have a tractable distribution T_α with domain $\mathsf{dom}T_\alpha = X_\alpha$, which can be sampled in polynomial time giving information about the universality of $\mathscr{L}(\alpha)$ in X_α.

So we consider a desired set of instances[4] α *with an associated family* $T = (T_\alpha)$ *of distributions that is tractable or polynomially samplable, and a subset operator* $\mathscr{L}_T$*, or just* $\mathscr{L}$*, such that* $\mathscr{L}(\alpha) \subseteq \mathsf{dom}T_\alpha$ *for all* α*.*

Definition 6.5. *A family of finite distributions* (T_α) *is called* polynomially samplable*, if there is a (randomized) algorithm* $\mathsf{select}_T(\alpha)$ *that selects an element from* T_α *in polynomial time w.r.t.* $|\alpha|$*.*

The index set of α's in the family (T_α) is simply the set of instances of the algorithm that involves sampling. A more customary definition of a polynomially samplable family of finite distributions is (T_i), for integers $i \geq 0$, such that the integer i is given in unary when complexity of sampling is considered [3]. However, Definition 6.5 is a more general version that works very well in our framework.

Definition 6.6. *A family of countable distributions* (T_α) *is called* tractable*, if there is a (randomized) algorithm* $\mathsf{select}_T(\alpha, \varepsilon)$*, where* $\varepsilon \in (0,1)$*, that is polynomial w.r.t.* $|\alpha|$ *and* $1/\varepsilon$ *and randomly selects an element from the truncated distribution* T^F *such that* $T(X - F) \leq \varepsilon$*, where* $X = \mathsf{dom}T_\alpha$ *and* F *is a set of* X*-elements determined by the algorithm once (*F *can be defined by a static variable, in the programming sense, so that different calls to* $\mathsf{select}_T(\alpha, \varepsilon)$ *yield the same value of* F*).*

The universality problem defined by T (and $\mathscr{L}$) is the language

$$U_T \;=\; \{\alpha \mid T(\alpha) = 1\},$$

where $T(\alpha)$ is a shorthand for the mass $T_\alpha\big(\mathscr{L}(\alpha)\big)$ of the subset $\mathscr{L}(\alpha)$ in $\mathsf{dom}T_\alpha$.

Remark 6.7. The non-universality problem defined by T and $\mathscr{L}$ is the language

$$\bar{U}_T \;=\; \{\alpha \mid T(\alpha) < 1\},$$

[4] Following the presentation style of [8, pg. 193] and [16], we refrain from cluttering the notation with the use of a variable for the set of instances α.

In [2], the authors also consider the emptiness problem $E_S = \{\alpha \mid S(\alpha) = 0\}$ and the non-emptiness problem $\bar{E}_S$, for some family of distributions S and subset operator $\mathscr{L}_S$. It turns out however that E_S is simply the universality problem U_T with respect to $\mathscr{L}_T$, where $\mathscr{L}_T(\alpha) = \overline{\mathscr{L}_S(\alpha)}$ and $T(\alpha) = 1 - S(\alpha)$. For this reason, in this paper we only consider the universality and non-universality problems.

Universality and non-universality problems can be hard. In [16], it is shown that even approximate versions of universality problems are hard. For any desired approximation tolerance $\varepsilon \in (0,1)$, the approximate universality problem is

$$U_{T,\varepsilon} = \{\alpha \mid T(\alpha) \geq 1-\varepsilon\},$$

and the approximate non-universality problem is

$$\bar{U}_{T,\varepsilon} = \{\alpha \mid T(\alpha) < 1-\varepsilon\},$$

As discussed in [16, 2], approximation alone, or randomization alone, cannot provide polynomial solutions, and that was the motivation for combining the two into PRAX algorithms. As stated in [8, pg. 417]: *"The answer to [what constitutes a "good" approximation] seems intimately related to the specific computational task at hand ... the importance of certain approximation problems is much more subjective ... [which] seems to stand in the way of attempts at providing a* comprehensive *theory of* natural *approximation problems."*

A PRAX algorithm for U_T is a randomized decision algorithm[5] $A(\alpha,\varepsilon)$ satisfying the following conditions:

1. if $\alpha \in U_T$ then $A(\alpha,\varepsilon) = \mathsf{True}$;
2. if $\alpha \notin U_{T,\varepsilon}$ then $\mathrm{P}[A(\alpha,\varepsilon) = \mathsf{False}] \geq 3/4$;
3. $A(\alpha,\varepsilon)$ works in polynomial time w.r.t. $1/\varepsilon$ and the size of α.

The main idea of the above definition is that, as both U_T and its approximate version can be hard, we accept as correct a yes answer to the universality of α, even if the mass of $\mathscr{L}(\alpha)$ in $\mathsf{dom}T_\alpha$ is ε-close to being universal, that is, $T\big(\mathscr{L}(\alpha)\big) \geq 1-\varepsilon$. The answer is incorrect exactly when the algorithm returns True and $T(\alpha) < 1-\varepsilon$, that is α is not even ε-close to being universal, but this happens with probability $< 1/4$, and in fact with probability $(1/4)^k$ if the algorithm is run k times.

A PRAX algorithm for $\bar{U}_T$ is a randomized decision algorithm $A(\alpha,\varepsilon)$ satisfying the following conditions:

1. if $\alpha \in \bar{U}_{T,\varepsilon}$ then $\mathrm{P}[A(\alpha,\varepsilon) = \mathsf{True}] \geq 3/4$;
2. if $\alpha \notin \bar{U}_T$ then $A(\alpha,\varepsilon) = \mathsf{False}$;
3. $A(\alpha,\varepsilon)$ works in polynomial time w.r.t. $1/\varepsilon$ and the size of α.

Definition 6.8. *A PRAX algorithm is a PRAX algorithm for a universality problem U_T or for a non-universality problem $\bar{U}_T$.*

As noted already in [17], any PRAX algorithm $A(\alpha,\varepsilon)$ for U_T can be turned to a PRAX algorithm $\bar{A}(\alpha,\varepsilon)$ for $\bar{U}_T$ by simply switching the values True and False in all the output (return) statements of $A(\alpha,\varepsilon)$.

[5] A decision algorithm halts on every input with the answer True or False.

References

[1] André Almeida, Marco Almeida, José Alves, Nelma Moreira & Rogério Reis (2009): *FAdo and Guitar: tools for automata manipulation and visualization*. In S. Maneth, editor: CIAA 2009: Fourteenth International Conference on Implementation and Application of Automata, 5642, Sydney, Australia, pp. 65–74, doi:10.1007/978-3-642-02979-0_10.

[2] Pantelis Andreou, Stavros Konstantinidis & Taylor Smith (2025): *Improved Randomized Approximation of Hard Universality and Emptiness Problems*. J. Automata, Languages and Combinatorics 30(1), doi:10.25596/jalc-2025-005.

[3] Sanjeev Arora & Boaz Barak (2009): *Computational Complexity – a modern approach*. Cambridge University Press, New York.

[4] Emmanuel Arrighi, Henning Fernau, Stefan Hoffmann, Markus Holzer, Ismaël Jecker, Mateus de Oliveira Oliveira & Petra Wolf (2021): *On the Complexity of Intersection Non-emptiness for Star-Free Language Classes*. CoRR abs/2110.01279. Available at `https://arxiv.org/abs/2110.01279`.

[5] Rafaela Bastos, Sabine Broda, António Machiavelo, Nelma Moreira & Rogério Reis (2016): *On the State Complexity of Partial Derivative Automata For Regular Expressions with Intersection*. In Cezar Câmpeanu, Florin Manea & Jeffrey O. Shallit, editors: Descriptional Complexity of Formal Systems - 18th IFIP WG 1.2 International Conference, DCFS 2016, Bucharest, Romania, July 5-8, 2016. Proceedings, Lecture Notes in Computer Science 9777, Springer, pp. 45–59, doi:10.1007/978-3-319-41114-9_4.

[6] Yuan Gao, Nelma Moreira, Rogério Reis & Sheng Yu (2017): *A Survey on Operational State Complexity*. J. Autom. Lang. Comb. 21(4), pp. 251–310, doi:10.25596/JALC-2016-251.

[7] Dora Giammarresi & Antonio Restivo (1997): *Two-dimensional Languages*. In Grzegorz Rozenberg & Arto Salomaa, editors: Handbook of Formal Languages, 3, pp. 215–267, doi:10.1007/978-3-642-59126-6_4.

[8] Oded Goldreich (2008): *Computational complexity - a conceptual perspective*. Cambridge University Press, doi:10.1017/CBO9780511804106.

[9] Solomon W. Golomb (1970): *A class of probability distributions on the integers*. J. Number Theory 2, pp. 189–192.

[10] Solomon W. Golomb (1992): *Probability, information theory, and prime number theory*. Discrete Math. 106/107, pp. 219–229.

[11] Bogdan Grechuk (2022): *On the smallest open Diophantine equations*. SIGACT News 53(1), pp. 36–57, doi:10.1145/3532737.3532743.

[12] Godfrey Harold Hardy & Edward Maitland Wright (1975): *An Introduction to the Theory of Numbers*, fourth edition.

[13] Michal Hospodár & Juraj Sebej (2026): *Square, power, positive closure, and complementation on ordered and star-free languages*. Theor. Comput. Sci. 1064, p. 115703, doi:10.1016/J.TCS.2025.115703.

[14] Jozef Jirásek & Galina Jirásková (2015): *On the boundary of regular languages*. Theor. Comput. Sci. 578, pp. 42–57, doi:10.1016/J.TCS.2015.01.022.

[15] Stavros Konstantinidis, António Machiavelo, Nelma Moreira & Rogério Reis (2021): *On the size of partial derivatives and the word membership problem*. Acta Informatica 58(4), pp. 357–375, doi:10.1007/S00236-021-00399-6.

[16] Stavros Konstantinidis, Mitja Mastnak, Nelma Moreira & Rogério Reis (2023): *Approximate NFA universality and related problems motivated by information theory*. Theor. Comput. Sci. 972, p. 114076, doi:10.1016/J.TCS.2023.114076.

[17] Stavros Konstantinidis, Nelma Moreira, Rogério Reis & Juraj Sebej (2024): *On the difference set of two transductions*. Theor. Comput. Sci. 1016, p. 114780, doi:10.1016/J.TCS.2024.114780.

[18] Markus Krötzsch, Tomás Masopust & Michaël Thomazo (2017): *Complexity of universality and related problems for partially ordered NFAs*. Inf. Comput. 255, pp. 177–192, doi:10.1016/J.IC.2017.06.004.

[19] Jing Lei & Joseph B. Kadane (2020): *On the Probability That Two Random Integers Are Coprime*. Statistical Science 35(2), pp. 272–279, doi:10.1214/19-STS737.

[20] Michael Mitzenmacher & Eli Upfal (2017): *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*, 2nd edition. Cambridge University Press.

[21] Rogério Reis & Nelma Moreira (Accessed in September of 2026): *FAdo: Tools for Formal Languages manipulation*. `http://fado.dcc.fc.up.pt/`.

[22] Taylor J. Smith (2021): *Closure, Decidability, and Complexity Results for Restricted Variants of Two-Dimensional Automata*. Ph.D. thesis, Queen's University. Available at `http://hdl.handle.net/1974/29180`.