# Towards a Computational Theory of Statistical Tests

Manuel Blum[*]
EECS Department - CS Div.
University of California
Berkeley, CA 94720, USA

Oded Goldreich[†]
Computer Science Department
Technion
Haifa, Israel.

TR-747, CSD, Technion, Israel (1992)

**Abstract**

We initiate a computational theory of statistical tests. Loosely speaking, we say that an algorithm is a *statistical test* if it rejects a "negligible" fraction of strings. We say that a statistical test is *universal* for a class of algorithms if it rejects all (but finitely many) of the strings rejected by each algorithm in the class.

We consider the existence and efficiency of universal statistical tests for various classes of statistical tests. We also consider the relation between ensembles passing statistical tests of particular complexity and ensembles which are indistinguishable from uniform by algorithms of the same complexity.

Some of our results refer to relatively simple statistical tests (e.g., those implemented by counter machines). In addition to their own merit, we hope that these results will stimulate investigations directed towards results that refer to more complex statistical tests (e.g., those implemented in log-space).

[*]Research was supported in part by NSF grant No. CCR88-13632.

[†]Research was supported in part by grant No. 89-00312 from the United States - Israel Binational Science Foundation (BSF), Jerusalem, Israel.

## 1 Introduction

In the first edition of Vol. 2, Knuth lists 8 empirical statistical tests. In the second edition, 11 tests are listed. There is no apparent rhyme or reason to the test selection, and future editions may well have even more tests. A comment, at the end of Section 3.3.2 in [10], asserts that at least some of the given statistical tests are "independent of the others" in the sense that there exists a sequence that passes all but the one test. Knuth offers no clues to a theory which may govern one's decision of which test one should use.

It would be desirable to have *one* "universal" statistical test with the following property. If a sequence fails *any* "reasonable" statistical test, be it a test in the 1st edition, 2nd edition, or any future edition of Knuth, then the sequence will fail the universal test. The question, of course, is what is a "reasonable" statistical test.

In this work we suggest to develop a theory of statistical tests based on the computational complexity of the tests (as algorithms).

### 1.1 Motivation

Statistical tests are the focus of considerable attention in many scientific disciplines. Hence, one may feel that there is little need to justify an attempt to present a theory of such tests. However, we feel that one should ask what are statistical tests good for, and in fact answering this question is a first step towards understanding the nature of these objects.

The need for statistical tests arises in a context where "randomness" is required (e.g., sampling procedures, randomized algorithms, randomized constructions). In such a context, "randomness" is supplied by a *random source* and statistical tests are used to check whether the source is "good". Why should one check the source? There are several reasons and they all boil down to the possibility that the source is faulty (and the need to detect such faults).

We first point out that one rarely has access to a "perfect" source (i.e., a source of independent and uniformly distributed random bits). What one *can* do is merely observe some physical phenomenon (e.g., a noise diode) which is considered to be "*somewhat* random". Typically, one converts, transforms, and/or expands measurements of the *physical phenomenon* into a *source* of "almost unbiased" bits. Faults may occur on several different conceptual levels:

- *wrong physical hypothesis*: The hypothesis concerning (the "randomness" of) the physical phenomenon may be wrong, and consequently the outcome of our source may be very far from uniform. In light of the difficulty of coming-up with reasonable hypotheses concerning the randomness of physical phenomena, a wrong hyothesis is quite likely.

- *wrong algorithmic assumptions*: Even if the hypothesis is correct, the transformation (of the physical phenomena into a bit sequence) may be wrong. Typically, the transformation consists of two phases. In the first phase the physical phenomena (being an analog signal) is converted into a bit sequence, whereas in the second phase this bit sequence is expanded into a much longer bit sequence (using a *pseudorandom generator*). In many cases these phases are implemented based on algorithmic assumptions which may turn out to be wrong. In particular, with respect to polynomial-time applications, the second phase can be performed only if one-way functions exist (a conjecture which is likely to remain an open questions for many years). Furthermore, it is likely that a pseudorandom generator used in practice will be based on either

1. a concrete and *stronger intractability assumption*, or
2. an *assumption restricting the type of application* (in which the random bit sequence is to be used).

Either type of assumptions are commonly made in order to allow more efficient expansion procedures (i.e., "pseudorandom generators"). In light of the poor understanding of such assumptions, wrong assumptions are quite likely.

- *programming bugs*: Finally, even if our physical hypothesis and algorithmic transformation are correct, the implementation of the transformation may contain a bug. Testing and/or checking a program that is supposed to produce a random output amounts to a statistical test!

What is needed is a test guaranteeing that if the source output passes the test then it (i.e., the source output) can be safely used in our application. Furthermore, it would be useful to have a single fast statistical test which accepts only bit sequences which can be safely used in any application belonging to some general "class" (of applications). Of particular interest is the class of all efficient (i.e., polynomial-time) algorithms, and the "simpler" subclass containing only log-space algorithms.

## 1.2 Statistical tests

Loosely speaking, a statistical test is an algorithm that accepts all but a negligible fraction of the strings. To convert this intuitive concept into a definition we need to say what is a negligible fraction. Following are three popular alternatives. A function $\mu(\cdot)$ is called *negligible* if

**N1)** $\mu$ is smaller than any constant (i.e., for every constant $\epsilon > 0$ and all sufficiently large $n$ it holds that $\mu(n) < \epsilon$). This interpretation is popular in statistics and other "classic" investigations.

**N2)** $\mu$ is smaller than any polynomial fraction (i.e., for every positive polynomial $p(\cdot)$ and all sufficiently large $n$ it holds that $\mu(n) < \frac{1}{p(n)}$). This interpretation seems most adequate in the context of efficient computation (specifically in the context of either polynomial-time or log-space algorithms).

**N3)** $\mu$ is exponentially decreasing (i.e., there exists $\epsilon > 0$ so that for all sufficiently large $n$ it holds that $\mu(n) < 2^{-\epsilon n}$). This interpretation is popular in some investigations in information theory and specifically in the area of classification theory.

The results of this paper are presented with respect to alternative (N2). The results apply also for *some* other definitions of negligence, and in particular the other alternatives presented above. Yet *not all* results hold with respect to *all* alternatives. Remarks concerning this issue are scattered throughout the paper.

## 1.3 Specific classes of statistical tests

We believe that two classes of statistical tests are of special interest. The class of (deterministic) polynomial-time statistical tests, and the class of log-space statistical tests. The importance of the first class is self-evident in light of the acceptable association of efficient procedures with polynomial-time algorithms. The interest in the second class is motivated by two facts. First, many applications in which randomness is used can be modelled by log-space machines. Secondly, all the "statistical tests" used in practice, and in particular all the tests mentioned by Knuth can be implemented by log-space machines.

Many of the results in this paper relate to two subclasses of the class of log-space statistical tests: the class of finite state machines, and the class of counter machines. The investigation of these classes may be considered a first step in an attempt to understand the class of log-space statistical tests. In addition, these classes are of some interest for their own right as they contain natural variants of all the tests mentioned by Knuth. We remark that the empirical tests presented by Knuth are not fully specified in the sense that the choice of the threshold (determining whether to reject or accept a sequence) is left open. Typically, setting the threshold to be a constant causes these tests to reject a constant fraction of the sequences, whereas setting the threshold to be linear (in the length of the sequence) causes these tests to reject an exponentially small fraction of the sequences as well as to be implemented by counter machines (see below).

## 1.4 Universal statistical tests

Loosely speaking, a statistical test is called *universal* for a particular class of statistical tests if it accepts only bit sequences which are accepted by all tests in the class. Universal tests promise to be very useful in practice, since they provide a way of replacing an infinite class of tests by a single test. The question, of course, is whether universal statistical tests exists with respect to interesting classes of tests, and furthermore whether such universal tests can be reasonably efficient.

### Some of our Results

We shall see in the sequel that one must allow such a universal test to use (slightly) more "resources" than any test in the class. In particular, no reasonable class of statistical tests, may contain a statistical test which is universal for the class. Furthermore, for sufficiently powerful classes (e.g., any class containing all on-line log-space tests), there must exist a gap between the rejection rate of the tests in the class and the rejection rate of any recursive statistical test that is universal for the class. We believe that it suffices to make the universal test slightly more "powerful" only in these two respects (i.e., higher computational complexity and higher rejection rate). For example, we present a quasi-polynomial-time universal test for the class of on-line logspace machines with exponentially decreasing rejection rate.

The universal test just mentioned (i.e., for the class of logspace tests) is neither "natural" nor "practical". We believe that presenting such a universal test is an important research goal.

> The Lempel-Ziv algorithm [11] may seem as a plausible candidate. Unfortunately, this algorithm is not universal for the class of log-space machines (since it accepts any de-Bruijn sequence [11, Thm. 5], whereas there exists de-Bruijn sequences that can be constructed and thus recognized and rejected by a log-space machine [4]). Yet, this example should at least give an idea of what is wanted.

We believe that the ability to present reasonable universal tests for a class of statistical tests is linked to a good understanding of the structure of the algorithms in the class. We were able to characterize the statistical tests in two natural subclasses of the class of logspace tests. In particular

- We present a necessary and sufficient condition for a finite state machine to be a statistical test. We use this condition to present a simple and natural test that is universal for the class of finite state statistical tests. These results hold for all alternatives of defining a negligible fraction.

- We present a necessary and sufficient condition for a weak counter machine (defined in section 5) to be a statistical test. The condition depends on the way one defines a negligible fraction. For alternatives (N2) and (N3) above, we present a simple and natural test which is universal for the class of statistical tests which are weak counter machines. For alternative (N1) we know only of a universal test which amounts to running all admissible tests.

We remark that the Lempel-Ziv algorithm [11] can be shown to be universal for the class of finite state tests (under alternatives (N1) and (N2)), and for the class of weak counter machines (under alternative (N2)). (The first statement is obvious whereas the second statement follows from the characterization of statistical tests which are weak counter machines.) However, the universal tests we present for these classes are much simpler. We also note that, under alternative (N1), the Lempel-Ziv algorithm is not universal for the class of weak counter machines.

### 1.5 Pseudorandomness and passing statistical tests

The ability of a probability ensemble to pass, with high probability, statistical tests of some complexity is even more useful in case such ability can be linked to pseudorandom properties of the ensemble. Loosely speaking, an ensemble is called pseudorandom with respect to some complexity class if no algorithm in the class can distinguish this ensemble from the uniform one. We stress that the distinguishing algorithm need not be a statistical test (e.g., it may reject half of the strings of each length).

**Some of our Results**

We consider the relation between the set of probability ensembles that are accepted by statistical tests of a particular complexity class and the set of ensembles which are pseudorandom with respect to arbitrary algorithms (not necessarily statistical tests) with the same complexity.

When putting no restrictions on the "algorithms" (i.e., allowing any function to be considered), there is a simple relation between these sets. We ask whether the same relation remains valid when restricting the class of algorithms. A positive result is presented for the class of constant-space on-line (i.e., finite state) machines, whereas a negative result is presented for the class of log-space on-line machines. Hence, much remains to be understood.

Finally, we use a construction of Levin [13] to prove that the existence of one-way function is a necessary condition for the existence of "non-trivial" ensembles (see section 7) which pass all probabilistic polynomial-time statistical tests. In light of the results in [1, 19, 7], this condition is also sufficient.

### 1.6 Organization

General definitions and observations are presented in sections 2 and 3, respectively. The other sections are devoted to specific classes of statistical tests. Section 4 deals with finite state machines, section 5 with (weak) counter machines, section 6 with (on-line) log-space machines, and section 7 with (probabilistic) polynomial-time machines.

## 2 Definitions

We start with the following standard conventions. By *negligible* we call any function $\mu : \mathbb{N} \mapsto \mathbb{N}$ which satisfies, for every constant $c$ and all sufficiently large $n$,

$$\mu(n) < \frac{1}{n^c}$$

By *non-negligible* we call any function $\mu : \mathbb{N} \mapsto \mathbb{N}$ which satisfies, for some constant $c$ and all sufficiently large $n$,

$$\mu(n) > \frac{1}{n^c}$$

(Note that a function may be neither negligible nor non-negligible, as the notion of non-negligible is a "strong negation" of the notion of negligible.)

**Remark 1** : The definition of negligence plays a pivotal role in the definition of a statistical test and in the definition of pseudorandomness. The above definition of negligence is particularly justified for the polynomial-time and/or logspace statistical tests and pseudorandomness. Yet, even in case the test is much weaker (or much stronger) it makes sense to define negligible probabilities this way since, in "natural" applications, the output of the test is latter used by a polynomial-time and/or logspace algorithm. Nevertheless, the theory can be developed using different notions of negligence. Most of the results presented in this work hold also for other natural definitions of negligence. Yet, some of the results (specifically, those concerning counter machines) are very sensitive to the definition of negligence (see remarks in Section 5).

**Notational Conventions:** By $X_n, Y_n, Z_n$ we denote random variables ranging over $\{0,1\}^n$. By $U_n$ we denote the random variable uniformly distributed over $\{0,1\}^n$. An *ensemble* is a sequence of random variables $\{X_n\}_{n\in\mathbb{N}}$, such that $X_n$ ranges over $\{0,1\}^n$.

## 2.1 Statistical Tests

**Definition 1** (acceptance/rejection of individual strings): *Let $A$ be a (probabilistic) algorithm.*

- *We say that $A$* accepts string *$\alpha$ (denoted $\overline{A}(\alpha) = \mathrm{ACC}$) if*

$$\mathrm{Prob}(A(\alpha)=1) \geq \frac{2}{3}$$

*where the probability is taken over $A$'s internal coin tosses.*

- *We say that $A$* rejects string *$\alpha$ (denoted $\overline{A}(\alpha) = \mathrm{REJ}$) if*

$$\mathrm{Prob}(A(\alpha)=1) \leq \frac{1}{3}$$

- *In case $\frac{1}{3} < \mathrm{Prob}(A(\alpha)=1) < \frac{2}{3}$ we say that $A$ is* undecided *about $\alpha$.*

Deterministic algorithms are never undecided (i.e. they either accept or reject each string). Also, algorithms which recognize some language with error probability which is bounded-away from one half (by some $\epsilon > 0$), can be easily modified into algorithms which are never undecided[1]. In general, it may be hard to get rid of the fact that a probabilistic algorithm is undecided on some strings without trivializing the algorithm (i.e., making it accept languages in an appropriate deterministic complexity class).

A statistical test is an algorithm which accepts all but a negligible fraction of the strings of a particular length.

**Definition 2** (statistical test): *We say that algorithm $A$ is a* statistical test *if the function $r(n) \stackrel{\mathrm{def}}{=} \mathrm{Prob}(\overline{A}(U_n) \neq \mathrm{ACC})$ is negligible. (Here the probability is taken over the probability space of $U_n$.)*

**Definition 3** (acceptance/rejection of ensembles):

- *We say that an ensemble $\{X_n\}_{n\in\mathbb{N}}$* is accepted by a statistical test *$T$ if $\mathrm{Prob}(\overline{T}(X_n) \neq \mathrm{ACC})$ is negligible.*

[1]The latter run the original algorithm $O(1/\epsilon)$ times, and take a majority vote.

- *We say that an ensemble* $\{X_n\}_{n\in\mathbb{N}}$ is rejected by a statistical test $T$ *if* $\text{Prob}(\overline{T}(X_n)\neq\text{REJ})$ *is negligible.*

Clearly, the uniform ensemble $\{U_n\}_{n\in\mathbb{N}}$ is accepted by all statistical tests. Of course, statistical tests may neither accept nor reject specific ensembles.

A statistical test is called universal *for some class* of statistical tests if it rejects all but a finite number of the strings rejected by each test in the class. The universal test need not be itself a member of the class (and in fact no "reasonable" class has a universal test residing inside the class).

**Definition 4** (universal statistical test): *A statistical test, $A$, is called* universal *for a class of statistical tests $\mathcal{T}$ if for every test $T$ in $\mathcal{T}$ and all sufficiently long $\alpha$, if $T$ rejects $\alpha$ then so does $A$ (i.e., the set $\{\alpha : (\overline{T}(\alpha) = \text{REJ}) \wedge (\overline{A}(\alpha) \neq \text{REJ})\}$ is finite).*

The definition of a universal statistical test encompasses two conflicting requirements. On one hand the universal test is *required to accept* all but a negligible fraction of the strings, whereas on the other hand it is *required to reject* all but finitely many of the strings rejected by any other test (in the class).

## 2.2 Computational Indistinguishability

Following are the two standard definitions of computational indistinguishability and pseudorandomness.

**Definition 5** [5, 19]: *The ensembles $\{X_n\}_{n\in\mathbb{N}}$ and $\{Y_n\}_{n\in\mathbb{N}}$ are* computationally indistinguishable by a class of (probabilistic) algorithms $\mathcal{A}$ *if for every algorithm, $A \in \mathcal{A}$, the difference*

$$|\text{Prob}(A(X_n)=1) - \text{Prob}(A(Y_n)=1)|$$

*is negligible.*

**Definition 6** [19]: *An ensemble is called* $\mathcal{A}$-pseudorandom *if it is computationally indistinguishable from the uniform ensemble by algorithms of the class $\mathcal{A}$.*

## 2.3 Domination

**Definition 7** [12]: *The ensemble $\{X_n\}_{n\in\mathbb{N}}$ is said to* (strongly) dominate *the ensemble $\{Y_n\}_{n\in\mathbb{N}}$ if the function $r(n) \stackrel{\text{def}}{=} \min_{\alpha\in\{0,1\}^n}\{\frac{\text{Prob}(X_n=\alpha)}{\text{Prob}(Y_n=\alpha)}\}$ is non-negligible.*

A more liberal notion of domination only requires that sets assigned negligible probability by the first ensemble are also assigned negligible probability by the second.

**Definition 8** : *The ensemble $\{X_n\}_{n\in\mathbb{N}}$ is said to* dominate *the ensemble $\{Y_n\}_{n\in\mathbb{N}}$ if for every sequence of sets $\{S_n\}_{n\in\mathbb{N}}$ if the function $p(n) \stackrel{\text{def}}{=} \text{Prob}(X_n \in S_n)$ is negligible so is the function $q(n) \stackrel{\text{def}}{=} \text{Prob}(Y_n \in S_n)$.*

Clearly, both notions of domination are reflexive and transitive.

## 3 General Observations and Problems

In this section we present general observations and problems concerning statistical tests. We first address the relation between the set of ensembles accepted by all statistical tests of particular complexity and the pseudorandom ensembles with respect to this complexity class. Next, we address the existence of universal statistical tests for various complexity classes.

To simplify the discussion, we assume without loss of generality, that all algorithms always output a Boolean value.

### 3.1 Pseudorandomness and Acceptance by Statistical Tests

Clearly, every pseudorandom ensemble is accepted by all statistical tests of the same complexity class. However, also ensembles which are not pseudorandom, for example ensemble $\{0U_{n-1}\}_{n\in\mathbb{N}}$, are accepted by all statistical tests. Furthermore,

**Proposition 1** : *Let $\{X_n\}_{n\in\mathbb{N}}$ be an $\mathcal{A}$-pseudorandom ensemble, for some complexity class $\mathcal{A}$, and let $\{Y_n\}_{n\in\mathbb{N}}$ be an ensemble which is dominated by $\{X_n\}_{n\in\mathbb{N}}$. Then, $\{Y_n\}_{n\in\mathbb{N}}$ is accepted by all statistical tests in $\mathcal{A}$. .*

**proof:** Suppose to the contradiction that $Y$ is not accepted by some test $A \in \mathcal{A}$. Hence the probability that $A$ does not accept $Y_n$ is not negligible. Since the probabilities assigned by $X_n$ are not negligibly smaller than those assigned by $Y_n$, it follows that the probability that $A$ does not accept $X_n$ is not negligible (– a contradiction). □

It is worthwhile noting that the ensembles satisfying the conditions of Proposition 1 may be far from pseudorandom. In particular,

**Observation 1** : Let $\mathcal{A}$ be a class of algorithms containing (but not necessarily equal to) the class of all finite state machines. Let $c > 0$ be an integer, and suppose that $\{X_n\}_{n\in\mathbb{N}}$ is a $\mathcal{A}$-pseudorandom ensemble. Then, the ensemble $\{X_{n-c}0^c\}_{n\in\mathbb{N}}$, which is not $\mathcal{A}$-pseudorandom, is accepted by all statistical tests in $\mathcal{A}$.

The proof is a simplification of the proof of the next observation.

**Observation 2** : Let $c > 0$ be a constant, and $\{X_n\}_{n\in\mathbb{N}}$ be a logspace-pseudorandom ensemble. (By logspace we mean the class of on-line algorithms which use logarithmic space.) Then, the ensemble $Y \stackrel{\text{def}}{=} \{X_{n-c\lceil\log_2 n\rceil}0^{c\lceil\log_2 n\rceil}\}_{n\in\mathbb{N}}$, which is not logspace-pseudorandom, is accepted by all logspace statistical tests.

**proof:** For simplicity let $c = 1$. To show that the ensemble $Y$ is not logspace-pseudorandom we use the following algorithm $D$. The algorithm counts (in binary) the number of bits read so far, denoted $m$, and keeps the last $k \stackrel{\text{def}}{=} \lceil\log_2 m\rceil$ bits read (note that $k$ is the current length of the binary counter). When reading the last bit of the input, algorithm $D$ outputs 0 iff all the last $k = \lceil \log_2 n \rceil$ bits are 0.

To establish that $Y$ is accepted by all logspace statistical tests we use a simulation argument. Suppose that $T$ is a logspace test which does not accept $Y$. We construct a logspace test $T'$ which does not accept $X$ reaching a contradiction to $X$'s pseudorandomness. Intuitively, on input $x$, the test $T'$ simulates the execution of $T$ on input $x0^{\log_2 |x|}$. This is done by initiating the execution of the test $T$ on input with prefix $x$. In "parallel", the test $T'$ computes $k$ satisfying $k = \lceil \log_2(|x| + k) \rceil$. When $T$ reaches the end of $x$, the test $T'$ appends $0^k$ at $x$'s ends and continues the execution of $T$. When $T$ stops after reading also these $k$ zero's, $T'$ stops with the same output. Hence, $T'$ does not accept $X$. It is left to show that $T'$ is a statistical test. To this end we use the fact that $T'(U_{n-k}) = T(U_{n-k}0^k)$ (where $k$ is $\approx \log_2 n$ as above) and hypothesis that $T$ is a statistical test. $\Box$

Hence, for all natural classes of algorithms, the set of ensembles which are accepted by statistical tests is much richer than the set of pseudorandom ensembles. An intriguing problem is whether this richness is captured by Proposition 1 (i.e., whether the accepted ensembles are exactly those dominated by pseudorandom ones). An affirmative answer is obtained in case one trivializes the notion of an algorithm as done in the next observation.

**Observation 3** : Let $\mathcal{F}$ be the class of all Boolean functions [2]. Then, $Y \stackrel{\text{def}}{=} \{Y_n\}_{n \in \mathbb{N}}$ is accepted by all statistical tests in $\mathcal{F}$ if and only if $Y$ is dominated by some $\mathcal{F}$-pseudorandom ensemble.

**Proof:** Using Proposition 1, it remains to show that if $Y$ is accepted by all statistical tests then it is dominated by some $\mathcal{F}$-pseudorandom distribution. We shall show that if $Y$ is accepted by all statistical tests (in $\mathcal{F}$) then $Y$ is dominated by the uniform ensemble. We prove this statement by proving the counter-positive.

Suppose that $Y$ is not dominated by the uniform ensemble. Then, using the liberal definition of domination, there exists a sequence of sets $\{S_n\}$ so that the function $\rho(n) \stackrel{\text{def}}{=} \text{Prob}(U_n \in S_n)$ is negligible while the function $\pi(n) \stackrel{\text{def}}{=} \text{Prob}(Y_n \in S_n)$ is not negligible. Consider the function $f \in \mathcal{F}$ defined by letting $f(\alpha) = 0$ if $\alpha \in S_n$ and $f(\alpha) = 1$ otherwise. Clearly, $f$ is a statistical test, yet $f$ does not accept the ensemble $Y$. The observation follows. $\Box$

Hence, if the set of ensembles accepted by statistical tests in $\mathcal{A}$ contains more than the ensembles dominated by $\mathcal{A}$-pseudorandom ones, then this is due to intrinsic algorithmic reasons and not to pure probabilistic ones. In section 4, we show that such "algorithmic reasons" are not supplied by finite automata. Namely, the ensembles accepted by finite state statistical tests are exactly those dominated by ensembles which are pseudorandom with respect to finite automata. On the other hand, in section 6 we show that there exists ensembles which are accepted by logspace statistical tests, and yet are not dominated by (logspace-)pseudorandom one. The problem remains open for other interesting complexity classes. In particular

[2]In this case, all "pseudorandom" ensembles are statistically close to the uniform one.

**Open Problem 1** : Do there exist ensembles accepted by polynomial-time statistical tests which are not dominated by polynomial-time-pseudorandom ensembles?

## 3.2 Universal Statistical Tests

One can show that, for any natural class of statistical tests, a test universal for the class can not be itself in the class. Namely,

**Theorem 1** *Let $\mathcal{A}$ be a class of algorithms satisfying the following three conditions:*

1. $\mathcal{A}$ is not trivial*: There exists $A \in \mathcal{A}$ so for every $\sigma \in \{0,1\}$ the set $\{\alpha : A(\alpha) = \sigma\}$ is infinite.*

2. $\mathcal{A}$ is closed under last bit omission*: For every $A \in \mathcal{A}$ there exists $A' \in \mathcal{A}$ so that, for every $\alpha \in \{0,1\}^*$ and $\sigma \in \{0,1\}$, it holds that $A'(\alpha\sigma) = A(\alpha)$.*

3. $\mathcal{A}$ is closed under binary operation*: For every $A_1, A_2 \in \mathcal{A}$ and every $b : \{0,1\}^2 \mapsto \{0,1\}$, there exists $A' \in \mathcal{A}$ so that for every $\alpha \in \{0,1\}^*$, it holds that $A'(\alpha) = b(A_1(\alpha), A_2(\alpha))$.*

*Then $\mathcal{A}$ contains no statistical test which is universal for $\mathcal{A}$.*

**proof:** Suppose on the contrary that $T \in \mathcal{A}$ is a statistical test which is universal for $\mathcal{A}$. Consider the algorithm $T'$ that, on input $\alpha\sigma$, outputs 0 if either $T(\alpha\sigma)$ or $T(\alpha)$ is 0. By conditions (2) and (3), $T'$ is also in $\mathcal{A}$. Furthermore, $T'$ is also a statistical test, since the number of strings not accepted by $T'$ is at most 3 times the number of strings not accepted by $T$. Yet, it can be shown that the universality of $T$ implies that the fraction of strings rejected by $T$ can be bounded below by a constant, and hence $T$ can not be a statistical test (in contradiction to our hypothesis). Details follows.

By condition (1), the class $\mathcal{A}$ contains a test, denoted $T_0$, that rejects infinitely many strings. Using the universality of $T$, with respect to the subclass containing $T'$ and $T_0$, it follows that there exists an integer $k$ so that the test $T$ rejects all strings of length $\geq k$ which are rejected by either $T'$ or $T_0$. It follows that there exists a string $\alpha$ (rejected by $T_0$) so that the test $T$ rejects $\alpha$ as well as all strings of length $\geq |\alpha|$ which are rejected by $T'$. By induction on the length of $\beta \in \{0,1\}^*$, it can be shown that $T$ must reject $\alpha\beta$ (since if $\alpha\beta$ is rejected by $T$ then $\alpha\beta\sigma$ is rejected by $T'$ and hence also by $T$). It follows that a constant fraction (i.e. at least a $\frac{1}{2^{|\alpha|}}$ fraction) of the strings are rejected by $T$, in contradiction to our hypothesis that $T$ is a statistical test. $\Box$

In the next two subsections we present universal statistical tests for two low complexity classes. These universal tests are quite efficient, although they are not in the corresponding classes. With respect to higher complexity classes, such as logspace, there exist no universal statistical tests at all. Namely,

**Theorem 2** *There exists no recursive statistical test that is universal for the class of logspace statistical tests.*

**proof:** Assume, to the contrary, that $T$ is a recursive statistical test that is universal for the class of logspace statistical tests. We reach a contradiction, by constructing a logspace statistical test $T'$, which rejects much more strings than $T$. The test $T'$ is constructed using traditional recursive theoretic techniques. Details follow.

Let $s(\cdot)$ denote an arbitrary space-constructible bound on the space used by algorithm $T$ (i.e., on each input of length $n$, algorithm $T$ uses at most $s(n)$ space). Let $f(n)$ denote the largest integer $m$ so that $m + s(m) \leq \log_2 n$. Intuitively, $f(n)$ represents the largest integer $m$ so that $T$ can be run on all $m$-bit strings within space bound $\log_2 n$.

Consider the standard lexicographic order of strings, and let $R_{n,k}$ denote the set of the first $\lceil \frac{2^n}{n^k} \rceil$ strings of length $n$. The algorithm $T'$ accepts a string $x$ if an only if $x \notin R_{|x|,k(|x|)}$, where the function $k(\cdot)$ is defined iteratively as follows. We say that $c > 0$ is *admissible for $n$* if for some $m \leq f(n)$, it holds that $k(m) = c - 1$ and the test $T$ rejects at most $\lfloor \frac{2^m}{m^c} \rfloor - 1$ of the strings of length $m$. In addition 0 is considered admissible for all $n$'s. We define $k(n)$ to be the largest integer $c$ that is admissible for $n$.

The reader can easily verify that the algorithm $T'$ works in logspace, that the function $k(\cdot)$ is non-decreasing, and that $k(n) \leq k(n-1)+1$ for all $n$'s. Furthermore, if $k(n) > k(n-1) = c$ then there exists an $m \leq f(n)$ so that $k(m) = c - 1$ (and thus the fraction of $m$-bit strings rejected by $T'$ is $\frac{1}{m^{c-1}}$), whereas the number of $m$-bit strings rejected by $T$ is at most a $\frac{2^m}{m^c} - 1$. Hence, if $k(\cdot)$ is unbounded then $T'$ rejects infinitely many strings not rejected by $T$ (since for each $c$ the $m$ mentioned above satisfies $\frac{2^m}{m^{c-1}} - (\frac{2^m}{m^c} - 1) \geq 1$). It is also easy to see that if the function $k(\cdot)$ is unbounded then $T'$ constitutes a statistical test. Hence, once we prove that the function $k(\cdot)$ is unbounded, we derive a contradiction (to the hypothesis that $T$ is universal for the class of logspace statistical tests).

Suppose to the contradiction that the function $k(\cdot)$ is bounded by the integer $c$, and let $N$ be an integer so that $c = k(N)$. By the hypothesis that $T$ is a statistical test, it follows that there exists an integer $M$ so that, for every $m \geq M$, the test $T$ rejects at most $\frac{1}{m^{c+1}}$ of the strings of length $m$. By definition of the function $k(\cdot)$ it follows that $k(n + s(n)) \geq c + 1$, for $n = \max\{N, M\}$, in contradiction to the hypothesis that $k(\cdot)$ is bounded by $c$. The theorem follows. $\square$

In light of the above result we must relax the requirements of universal statistical tests for complexity classes such as logspace and higher. One possible direction is to establish a "gap" between the accepting rate of the algorithms in the class and the accepting rate required of the universal algorithm. Two approaches are possible:

1. Allow the universal algorithm to reject a *small yet non-negligible* fraction of the strings. The rejection rate of such universal algorithms can be specified. For example, one can present a universal algorithm for the class of probabilistic polynomial-time statistical tests so that this universal algorithm rejects a $\frac{1}{n^3}$ fraction of the strings of length $n$.

2. Investigate universal tests for subclasses of statistical tests which have a *specified* negligible rejection rate. For example, one can present a universal statistical test for the class of probabilistic polynomial-time statistical tests which accept all but an exponential fraction of all strings (i.e., the class contains algorithms which accept all but an exponential fraction, whereas the universal algorithm accept all but a negligible fraction of the strings). More generally, for any negligible function $\mu$, there exists a universal statistical test for the class of probabilistic polynomial-time (resp., log-space) algorithms which accept all but at most a $\mu(n)$ fraction of the strings of length $n$. In case $\mu$ is polynomial-time computable, the universal test for the polynomial-time class can be implemented in probabilistic super-polynomial-time (e.g., time $n^{\log_2^* n}$). An analogous result for logspace is presented in Subsection 6, with the advantage that the universal test is deterministic.

We prefer the second approach.

## 4 Finite Automata as Statistical Tests

The simplest (and most restricted) class of algorithms we consider are finite automata. According to Definition 2, we say that a finite automaton, $A$, is a statistical test if the fraction of strings of length $n$ that $A$ rejects[3] is negligible. We start by characterizing the set of finite state statistical tests and then propose a statistical test (outside this set) which is universal for this set.

### 4.1 Characterization of Finite State Statistical Tests

We consider the directed graph (*digraph*), $G$, defined by a finite automaton $A$ (ignoring the binary labelling of edges). Note that $G$ may have parallel, anti-parallel and self-loop edges. We consider the strongly connected components[4] of the digraph $G$. Let $\{C_i : i \in I\}$ be the partition of $G$'s vertices to strongly connected components. We consider the directed acyclic graph (*dag*), $D_G$, in which the $C_i$'s are vertices and there is an edge from $C_i$ to $C_j$ if and only if there exists $u \in C_i$, $v \in C_j$ so that $(u, v)$ is an edge in $G$. A *sink* is a strongly connected component which has outdegree zero in this dag. Using the definition of a statistical test it follows that

**Proposition 2** *Let $A$ be a finite state statistical test. Then the states appearing in its sinks must be accepting. Furthermore, there exists a finite state statistical test $A'$ which is identical to $A$ except (possibly) that the states of $A'$ which are not in a sink are rejecting.*

**proof sketch:** We consider a random walk of length $n$ on the digraph $G$ (associated with the automaton $A$) starting in the vertex corresponding to $A$'s initial state (this walk represents the computation of $A$ on a random $n$-bit input string).

[3] Since $A$ is deterministic, it is never undecided.

[4] A digraph is called strongly connected if there is a directed path between every (ordered) pair of vertices. The strongly connected components of a digraph are the maximal sets of vertices inducing a strongly connected sub(di)graph.

Let $v$ be a vertex in a sink $S$. With constant probability (i.e., at least $2^{-q}$ where $q$ is the number of states in $A$) this walk enters $S$ after a constant number of steps (i.e. $q$). From that point on, the walk never leaves $S$. The "location" of the walk after $m \geq n-q$ steps is a random variable, denoted $P_s(m)$, depending on $s$ the first vertex of $S$ reached in the $n$-step walk. The behaviour of $P_s(m)$ converges to a random variable $Q(m \bmod k)$, where $k$, $1 \leq k \leq q$ is an integer (see [8] proof of Thm. 4.1.4 which refers to the convergence of a regular Markov Chain to the stationary distribution). The rate of convergence is exponential in $\frac{m}{k}$, and for some $i$ $(1 \leq i \leq k)$ the probability that $Q(i)$ equals $v$ (in which we are interested) is a positive constant depending only on $A$ (and again being at least $2^{-q}$). Hence, for $n \equiv i \pmod{k}$, a random walk of length $n$ ends in $v$ with constant probability. It follows by Definition 2 that $v$ must be accepting.

Using an even simpler argument (cf. [8] proof of Thm. 3.1.1) it follows that the probability that a $n$-long random walk does not end in a sink is bounded above by $2^{-n/q}$. Hence, all non-sink vertices may be made rejecting without violating the conditions of Definition 2. $\square$

**Remark 2** : By the above proof it follows that the characterization of finite state statistical tests is not effected if the definition of negligible is changed either to "smaller than any constant" or to "exponentially small". Hence, also the results of the next two subsections remain valid also if the definition of negligence is changed to "smaller than any constant". More generally, all the results hold as long as negligible is defined as smaller than any function in a recursively enumerable set of functions which do not have a maximum element (i.e., a function in the enumeration which is greater/equal than all other functions on all but finitely many points).

## 4.2 A Universal Test for the Class of Finite Automata

Following is a description of a very simple test which is universal for the class of finite state statistic tests. Let $k : \mathbb{N} \mapsto \mathbb{N}$ be any easy to compute function which is both monotonically non-decreasing, unbounded (by a constant) and bounded above by the function $\log_2 n - 3\log_2\log_2 n$. The test checks whether all $k(n)$-bit long strings appear as substrings in the $n$-bit long input. Namely, the test accepts the string if and only if all possible $k(n)$-bit long substrings appear in it. We call this test the *occurrence test.*

**Theorem 3** *The occurrence test is universal for the class of finite state statistical tests.*

**proof sketch:** Let $T$ be an arbitrary finite state statistical test. A *homing sequence* for $T$ is any string $\alpha$ satisfying the following: for every state $s$ of $T$ an execution starting at state $s$ and following the edges labelled by $\alpha$ ends in a sink of $T$. It is well known that every finite automaton with $q$ states has a homing sequence of length $q^2$. Hence, sufficiently long strings accepted by the occurrence test must contain the homing sequence of $T$ as a substring, and thus are also accepted by $T$. Finally, it is left to show that the occurrence test is indeed a statistical test. This is done by a straightforward calculation of the probability that some $k$-bit long string does not appear as substring in a uniformly chosen $n$-bit long string. This probability is easily bounded above by

$$(1-\frac{1}{2^k})^{\frac{n}{k}} \cdot 2^k < 2^{-\frac{n-k^2\cdot 2^k}{k\cdot 2^k}}$$

The proposition follows. $\square$

Clearly, the occurrence test cannot be implemented by a finite automata. Furthermore, using Theorem 1, it is easy to see that no finite automata can be universal for the class of finite state statistical tests.[5] In particular, if the function $k$ is chosen to be a constant then the resulting test (is a finite automata but) is not universal for the class.

## 4.3 Pseudorandomness and Acceptance by Statistical Tests

By Proposition 1, all ensembles dominated by (Finite Automata)-pseudorandom ensembles are accepted by all finite state statistical tests. We now show that these are all the ensembles which are accepted by all finite state statistical tests. Namely,

**Theorem 4** *An ensemble is accepted by all finite state statistical tests if and only if it is dominated by an ensemble that is pseudorandom with respect to all finite automata.*

**proof:** Let $X \overset{\text{def}}{=} \{X_n\}_{n\in\mathbb{N}}$ an ensemble which is accepted by all finite state statistical tests. We will construct an ensemble $Y \overset{\text{def}}{=} \{Y_n\}_{n\in\mathbb{N}}$ and show that for each $i$ the function $g_i : \mathbb{N} \mapsto \mathbb{N}$, defined by $g_i(n) \overset{\text{def}}{=} |\text{Prob}(A_i(Y_n)=1) - \text{Prob}(A_i(U_n)=1)|$ where $A_i$ is the $i^{\text{th}}$ automaton (in a standard enumeration of finite automata), is negligible. The theorem will follow.

The idea behind the construction of the distributions $Y_n$ is to rescale the distributions $X_n$ so that they exhibit the same statistic profile as the uniform ones with respect to leading a finite automata to terminate at a particular state residing in its sink. The rescaling satisfies the domination condition and guarantees that no finite automata can distinguish $Y_n$ from uniform. The part of the distribution which does not lead an automata to a sink remains unchanged. Using the fact that $X_n$ is accepted by all finite state statistical tests, this later part is shown to carry a negligible probability mass. Details follow.

For every integer $k$, let $P_k$ be the cross-product of the first $k$ finite state machines. Fixing $k$, we consider, for every integer $m$, the execution of $P_k$ on input $X_m$ (i.e., we consider a random variable with sample space $X_m$). Consider a modification of $P_k$ in which all sink states are accepting and all other states are rejecting. Since this modification is a finite state statistical test which accepts the ensemble $X$, it follows that the probability that the computation of $P_k$ on input $X_m$ does not terminate in a sink is a negligible function (in $m$). Let $m_k$ be the largest integer, $m$, for which the probability that the computation of $P_k$ on input $X_m$ does not terminate in a sink is greater than $\frac{1}{m^k}$ (by the previous assertion this integer does exist!). Let $M_k \overset{\text{def}}{=} \max\{m_j : j \leq k\}$. Clearly, the sequence $M_1, M_2, \ldots$ is non-decreasing. Define a function $k : \mathbb{N} \mapsto \mathbb{N}$ so that $k(n) = \min\{\sqrt{\log_2 \log_2 n}, \max\{j : M_j < n\}\}$. Clearly the function $k$ is both

[5]An alternative proof proceeds as follows. Consider a finite automata $A$ with $q$ states, and let $\alpha \in \{0,1\}^q$ be a string "leading" $A$ from its initial state to any of its sink. If $A$ is a statistical test then, by Proposition 2, all its sinks must be accepting. Hence, $A$ must accept all strings of the form $\alpha^n$. Such strings are rejected by a finite state statistical test with $q+1$ states, hence $A$ cannot be universal for the class of finite automata statistical tests.

monotonically non-decreasing, unbounded (by a constant) and bounded above by the function $\sqrt{\log_2 \log_2 n}$.

For each state, $s$, define the set of inputs $I_n(s) \subseteq \{0,1\}^n$ so that on input $\alpha \in I_n(s)$ automata $P_{k(n)}$ terminates in state $s$. Let $x_n(s) \stackrel{\mathrm{def}}{=} \mathrm{Prob}(X_n \in I_n(s))$ and $X_n(s)$ be $X_n$ conditioned on it being in $I_n(s)$ (i.e., $\mathrm{Prob}(X_n = \alpha) = x_n(s) \cdot \mathrm{Prob}(X_n(s) = \alpha)$). Let $u_n(s) \stackrel{\mathrm{def}}{=} \mathrm{Prob}(U_n \in I_n(s))$ (i.e., the probability that on uniformly chosen input the automata terminates in state $s$). Let $x_n$ (resp. $u_n$) denote the sum of the $x_n(s)$'s (resp. $u_n(s)$'s) taken over states $s$ residing in the sinks of $P_{k(n)}$. We are now ready to define the distribution $Y_n$. Define $Z_n$ so that $\mathrm{Prob}(Z_n = s) = x_n \cdot \frac{u_n(s)}{u_n}$, for each state $s$ residing in a sink of $P_{k(n)}$, and $\mathrm{Prob}(Z_n = s) = x_n(s)$ otherwise (i.e., for each $s$ not residing in a sink). Set $Y_n = X_n(Z_n)$.

We first argue that $Y_n$ dominates $X_n$. Clearly, for any state $s$ not residing in a sink of $P_{k(n)}$, we have $\mathrm{Prob}(Z_n = s) = x_n(s)$ and $\mathrm{Prob}(Y_n = \alpha) = \mathrm{Prob}(X_n = \alpha)$, for every $\alpha \in I_n(s)$. Also, for any state $s$ residing in a sink of $P_{k(n)}$, we have $\mathrm{Prob}(Z_n = s) = x_n \cdot \frac{u_n(s)}{u_n}$ and

$$\mathrm{Prob}(Y_n = \alpha) = (x_n \cdot \frac{u_n(s)}{u_n}) \cdot \mathrm{Prob}(X_n(s) = \alpha) \geq (x_n \cdot \frac{u_n(s)}{u_n}) \cdot \mathrm{Prob}(X_n = \alpha)$$

for every $\alpha \in I_n(s)$. Clearly, $\frac{x_n}{u_n} > \frac{1}{2}$ (since $x_n > 1 - \frac{1}{n^{k(n)}}$ and $u_n \leq 1$). By the proof of Proposition 2, it follows that for every $s$ residing in a sink $u_n(s) > 2^{-2q}$, where $q < k(n)! < \frac{1}{2} 2^{k(n)^2}$ is the number of states in $P_{k(n)}$. Using $k(n) < \sqrt{\log_2 \log_2 n}$, we get $x_n \cdot \frac{u_n(s)}{u_n} > \frac{1}{2n}$, and domination follows.

We now turn to show that $Y_n$ is pseudorandom with respect to all finite state machines. We show that for every finite state machine, $A_i$, and every constant $c$, the function $g_i$ defined above (as the "distinguishing gap" of $A_i$) is bounded above by the function $\frac{1}{n^c}$ (i.e., for all sufficiently large $n$: $g_i(n) < \frac{3}{n^c}$). Let $N$ be an integer satisfying $k(n) \geq i$ and $k(n) \geq c$, for each $n > N$ (by definition of the function $k$ such $N$ necessarily exists). Clearly, the computation of $A_i$ on input of length $n > N$ is reflected by the computation of the product automata $P_{k(n)}$ (on inputs of length $n$). Let $y_n(s)$ denote the probability that on input $Y_n$ machine $A_i$ terminates in state $s$ (i.e., $y_n(s) = \mathrm{Prob}(Z_n = s)$). For each $n > N$, and every state $s$ residing in a sink (of $P_{k(n)}$), $y_n(s) = \frac{x_n}{u_n} \cdot u_n(s)$. Using the fact that $n > m_c$ it follows that $x_n > 1 - \frac{1}{n^c}$. On the other hand, by the proof of Proposition 2, $u_n > 1 - 2^{-\frac{n}{q}}$ (with $q < \frac{1}{2} 2^{k(n)^2}$). It follows that, for every state $s$ residing in a sink of $P_{k(n)}$ and all sufficiently large $n$, we have $|u_n(s) - y_n(s)| < \frac{u_n(s)}{n^c}$. Furthermore, using the above mentioned bounds on $x_n$ and $u_n$, the probability that $A_i$ terminates in a non-sink on input drawn from either $Y_n$ or $U_n$ is smaller than $\frac{1}{n^c}$. It follows that, for all sufficiently large $n$, the sum of $|u_n(s) - y_n(s)|$ taken over all states $s$ not residing in a sink of $P_{k(n)}$ is bounded above by $\frac{2}{n^c}$. Hence, $g_i(n) \leq \sum_s |u_n(s) - y_n(s)| < \frac{3}{n^c}$, and the theorem follows. $\square$

## 5 Counter Machines as Statistical Tests

We now consider a wider class of tests. The algorithms we consider are counter machines with on-line access to the input. (Hence, these machines are not equivalent to Turing machines.) We start by considering machines with one counter. Once a new input symbol (i.e., bit) is read, and depending on its current state and on whether the counter is empty or not, the machine may change its state and increment/decrement its counter by a fixed number of units. Allowing the counter to assume negative values does not add to the power of such machines (since the sign may be encoded by the state).

To simplify the analysis even further, we consider only *weak 1-counter machines*. In these machines, the next transition is not effected by whether the counter is empty or not. Hence, we must allow the counter to assume negative values (otherwise the power of this model is further reduced). On the other hand, the final verdict ("accept" or "reject") may depend on the sign of the counter (i.e., negative, zero, or positive). Again, as in previous section, the machines are deterministic and hence are never undecided about an input. We start by characterizing the set of weak 1-counter statistical tests, and next propose a statistical test (outside this set) which is universal for this set.

## 5.1 Characterization of Weak 1-Counter Statistical Tests

We consider again the directed graph (*digraph*), $G$, defined by a finite automaton $A$. This time the edges are labelled by binary input symbols and by the corresponding counter modifications (which without loss of generality are 0, +1 or −1).

The characterization of weak 1-counter statistical test consists of a characterization of the accepting *configurations*. The possible configurations at the end of the computation are pairs, $(s, \sigma)$, where $s$ is a state of the finite control and $\sigma \in \{\mathrm{P}, \mathrm{Z}, \mathrm{N}\}$ indicate the sign of the counter (i.e., $\sigma = \mathrm{P}$ indicates positive counter value, $\sigma = \mathrm{Z}$ indicates zero counter value, and $\sigma = \mathrm{N}$ indicates negative counter value).

The characterization of configurations depends not only on the component structure of the digraph $G$, but also on the expected value of the counter modification in a random walk inside each sink of $G$. To this end we recall that a random walk in each sink defines an ergodic Markov Chain. Assume for a moment that this Markov Chain is regular (i.e., that the gcd of set of legths of all directed circuits in $G$ is 1), and let $\pi$ be the stationary distribution of this chain. Then, the *average of a sink* is defined as $\sum_e \pi(e)m(e)$, where $\pi(e)$ is the probability of passing on edge $e$ and $m(e)$ is the counter modification on this edge. In general, the Markov Chain may be cyclic (i.e., the gcd of cycle lengths is $d > 1$) meaning that only for some $d > 1$ (but $d \leq q$) making $d$ random steps in a row yields a regular Markov Chain. In this general case, $d$ different "limit distributions" are defined (depending on the equivalent class of the starting vertex) and the average of the sink is defined as the average of the expected counter modifications for each of the corresponding limit distributions.

A configuration is called *reachable* if it can be reached in a computation on some input.

**Proposition 3** *Let $A$ be a weak 1-counter statistical test, and $(s, \sigma)$ be an (end) configuration. Then, the configuration must be accepting if one of the following conditions hold:*

1. *$s$ resides in a sink with positive average and $\sigma = \mathrm{P}$.*

*2. $s$ resides in a sink with negative average and $\sigma = \mathrm{N}$.*

*3. $s$ resides in a sink with zero average, and $(s, \sigma)$ is a reachable configuration.*

*Furthermore, there exists a weak 1-counter statistical test $A'$ which is identical to $A$ except (possibly) that all the configurations not mentioned in items (1-3) are rejecting.*

**proof sketch:** As in the proof of Proposition 2 we show that for $n \equiv i \pmod{k}$, $k < q$, and every state $s$ residing in a sink, a random walk of length $n$ terminates at $s$ with constant probability. The question is what is the value of the counter at termination. For states $s$ residing in sinks in which all cycles have zero sum the value in the counter equals its value at the first time during computation in which $s$ is visited. Hence, reachable $(s, \cdot)$ configurations, for $s$ is such a sink, are reached with constant probability, at the end of the computation. It is also clear that configurations which are not reachable can be made rejecting without affecting the acceptance profile of the machine. It is left to deal with sinks in which not all cycles have zero sum.

Using the Central Limit Theorem for Markov Chains (cf. [8] Thm. 4.6.9) it follows that the sum of counter modifications in a random walk is unlikely to deviate linearly from the expectation (i.e., the CLT guarantees that such deviation occurs with probability tending to zero). Hence, items (1) and (2) above follow. For the justification of (the rest of) item (3) we rely on a Local Limit Theorem for Markov Chains (cf. [6] Section 20 for a general exposition and [17] for a result implying what we need). The Local Limit Theorem guarantees that the probability that the sum of counter modifications in an $n$-step long random walk equals its expectation, up to an additive term of $q$, is $\Theta(1/\sqrt{n})$. Hence, item (3) follows. To prove the "furthermore clause" referring to items (1) and (2), we need a Large Deviation Theorem for Markov Chains (cf. [3] Section 2.4). This theorem guarantees that only with exponentially small probability the sum of counter modifications in a random walk may deviate linearly from its expectation[6]. Hence, the "furthermore" clause follows and so does the entire proposition. $\Box$

**Remark 3** : Unlike the situation with finite state statistical tests, the characterization does depend on the particular definition of a negligible fraction. In particular, if negligible is interpreted as "smaller than any constant" then $(\cdot, \mathrm{Z})$ configurations need not be accepting (and may without loss of generality be rejecting).

[6]Actually the quoted result can be derived using more elementary methods as follows. Partition the random walk to sufficiently long blocks and consider the average of these short random walks which start at arbitrary states, as done in the proof of Theorem 5. To each block associate a random variable representing the counter modification in a random walk along the block starting in the worst possible state. These random variables have expectation close to the average of the sink (defined as expected counter modification with respect to the limit distributions associated with the sink). Now analyze the sum of these random variables using Chernoff bound. The application of Large Deviation Theorem yields a better constant in the exponent as well as a matching lower bound, both irrelevant for our purposes.

## 5.2 A Universal Test for the Class of Weak 1-Counter Machines

Following is a description of a very simple test which is universal for the class of finite state statistic tests. Let $k : \mathbb{N} \mapsto \mathbb{N}$ be any easy to compute function which is both monotonically non-decreasing, unbounded (by a constant) and bounded above by the function $\log_2 n - 5 \log_2 \log_2 n$. The test partitions the input string $\alpha$ into $\frac{|\alpha|}{k(|\alpha|)}$ blocks of length $k(\alpha|)$ each, and checks whether all $k(|\alpha|)$-bit long strings appear approximately as frequently in these blocks. More specifically, the frequency test computes, $\pi_\beta(\alpha)$ which equals $|\{i : \beta = \alpha_i\}|$, where $k \stackrel{\rm def}{=} k(|\alpha|)$ and $\alpha = \alpha_1 \cdots \alpha_{n/k}$ with $\alpha_i \in \{0,1\}^k$. The test accepts $\alpha$ if and only if $|\frac{\pi_\beta(\alpha)}{n/k(n)} - \frac{1}{2^{k(n)}}| < \frac{1}{k(n)} \cdot \frac{1}{2^{k(n)}}$ for all $\beta \in \{0,1\}^{k(n)}$. We call this test the *frequency test.*

**Theorem 5** *The frequency test is universal for the class of weak 1-counter statistical tests.*

**proof:** Let $T$ be an arbitrary weak 1-counter statistical test with $q$ states. We start by proving the following

> *Claim:* For every $\epsilon > 0$ and all sufficiently long $\alpha$, if $\alpha$ is accepted by the frequency test then the test $T$ enters a sink after reading at most $\epsilon|\alpha|$ bits of its input.
>
> *proof:* Let $n = |\alpha|$. Assume on the contrary that the test $T$ does not enter a sink after reading an $\epsilon$ fraction of the input bits. Repeating the argument used in the proof of Theorem 3, it follows that this fraction of the input does not contain an occurrence of a homing sequence for $T$. Let us denote by $\gamma$ any $q^2$-bit long homing sequence of $T$, and $h \stackrel{\rm def}{=} |\gamma|$. We first compute the fraction of $k(n)$-bit blocks, in a $n$-bit long random string, which do not contain $\gamma$ as a substring. This fraction is certainly bounded by $(1 - 2^{-h})^{k(n)/h} < 2^{-k(n)/h^2}$. Hence, the fraction of such blocks in $\alpha$ (which is accepted by the frequency test) is at most $(1 + \frac{1}{k(n)}) \cdot 2^{-k(n)/h^2}$. This fraction is smaller than any constant $\epsilon$ and hence it cannot be the case that $\gamma$ does not appear in the $\epsilon n$ prefix of $\alpha$.

Let $\alpha$ be a sufficiently long string passing the frequency test. We consider the computation of $T$ on input $\alpha$ from the moment in which $T$ entered a sink (which by the above Claim happens after making at most $\epsilon n$ steps). If all of $T$'s sinks have average (counter modification) zero then a computation of $T$ which enters a sink always terminates in an accepting configuration (see item (3) in the characterization). Similarly, we can discard all sinks with zero average, and concentrate on sinks with non-zero average. Let $\sigma$ be the minimum deviation from zero of the average of any non-zero sink of $T$ (i.e. of either positive or negative average). The Central Limit Theorem for Markov Chains (cf. [8] Thm. 4.6.9) guarantees that for any $p$ and for sufficiently large $K$ (depending only on the sink of $T$) the probability that a random walk of length $K$ starting from any location in the sink yields total counter modification of less than $\frac{\sigma K}{2}$ (towards the expected direction) is at most $p$. Let $c \geq 1$ be the maximum counter modification in a single transition of $T$. Then for sufficiently large $K$ a random walk of length $K$ starting from any location in the sink has expected counter modification of at least $\frac{\sigma K}{2} - pcK > \frac{\sigma K}{4}$ (as $p$ can be chosen small enough). Recall that for sufficiently large $n$, indeed $k(n)$ is greater than the minimal $K$ required above. Set $\epsilon < \frac{1}{16c} \cdot \sigma$. By the above Claim, it is guaranteed that the computation of $T$ on an $n$-bit string, $\alpha$, accepted by the frequency test enters a sink of $T$ after at most $\epsilon n$ steps. If this is a zero-average sink - we are done. Otherwise, assume without loss of generality that we entered a negative-average sink. Clearly, upon entering the sink there are at most $c\epsilon n$ units in the counter. We will show next, that in the remaining $(1-\epsilon)n > \frac{n}{2}$ steps the counter is decreased so that at termination it holds a negative value. Using item (2) of the characterization it follows that $T$ accepts the string $\alpha$.

To show that in the remaining steps the counter is decreased by at least $\epsilon cn$ units, we first assume that in these steps all $k(n)$-bit long substrings appear with equal frequency. By the above, we are guaranteed that the average counter decrease per $k(n)$-bit long block is at least $\frac{\sigma k(n)}{4}$. Hence, in $(1-\epsilon)n > \frac{n}{2}$ steps of equal block frequency we get a decrease of at least $\frac{\sigma n}{8}$. However, the frequencies of occurrence of the various the $k(n)$-bit long substrings may deviate from being equal. Yet, the maximum possible deviation is bounded by $\max\{\epsilon, \frac{1}{k(n)}\}$, which for sufficiently large $n$ yields a bound of $\epsilon cn$ on the possible deviation of the sum of modifications. Hence, in the $(1-\epsilon)n$ steps we get a decrease of at least $\frac{\sigma n}{8} - \epsilon cn > \epsilon cn$ (since $\epsilon < \frac{1}{16c} \cdot \sigma$).

Finally, it is left to show that the frequency test is indeed a statistical test. This is done by application of Chernoff bound. The probability that a particular $k(n)$-bit string appears in $t$ blocks, where $t \notin [(1-\frac{1}{k(n)}) \cdot \frac{n}{k(n)2^{k(n)}}, (1+\frac{1}{k(n)}) \cdot \frac{n}{k(n)2^{k(n)}}]$, is bounded above by an exponential in $-O(\frac{n}{k^3 2^k})$. The proposition follows. $\Box$

**Remark 4** : The frequency test cannot be implemented by a weak 1-counter machine. More generally, by Theorem 1, the class of weak 1-counter machines does not have a universal test which resides in the class itself. Also, setting the function $k$, in the definition of the frequency test, to be a constant yields a test which is not universal for the class of weak 1-counter machines.

**Remark 5** : Under an appropriate choice of parameters, it can be shown that a test based on the Lempel-Ziv algorithm [11] dominates the frequency test (i.e., rejects all the strings that are rejected by the frequency test). Hence, the LZ algorithm can be viewed as a universal statistical test for the class of weak 1-counter machines.

**Remark 6** : Under the liberal interpretation of "negligence" (by which any non-increasing function tending to zero is negligible), the frequency test is not universal for weak 1-counter machines which are statistical tests. Augmenting the frequency test so that it is universal also under the liberal interpretation of "negligence" boils down to an algorithm that is very close to one that runs all possible statistical tests in the class. We also mention that under the liberal interpretation of "negligence" the Lempel-Ziv algorithm is not universal either. In particular the LZ algorithm accepts de-Bruijn sequences whereas such sequences can be rejected by a weak 1-counter machine which compares the number of 1's to the number of 0's (and rejects the string if their difference is zero).

## 5.3 Extensions

We believe that the results of the two previous subsections can be extended to weak counter machines possessing *any constant number of counters*. The difficult is *not* in the algorithmic part, but rather in our knowledge of probability theory (specifically Local Limit Theorems for Markov Chains). To be more specific, we need a Local Limit Theorem referring to a finite Markov Chain with states associated with $d$-dimensional vectors, and to the (vector) sum of the states visited in an $n$-long random walk on the chain. We believe that, under some non-degeneracy conditions, the probability that the sum over such a walk equals the expect value (which is a $d$-dimensional vector) is related to $\frac{1}{n^{\frac{d}{2}}}$. We refer to this conjecture as $\mathcal{LLT}(d)$. We point out that $\mathcal{LLT}(1)$ is known to hold, and this fact has been used in subsection 5.1. However, we failed in our attempts to trace such a proof of $\mathcal{LLT}(d)$, for $d > 1$. Hence, all we can say is

**Theorem 6** *Suppose that for some $d > 1$, conjecture $\mathcal{LLT}(d)$ holds. Then the frequency test, presented in subsection 5.2, is universal for the class of weak $d$-counter statistical tests.*

We believe that the frequency test is universal also with respect to the class of (general) $d$-counter statistical tests, for every $d \geq 1$. Proving such a result, even for $d = 1$ requires a much more cumbersome characterization. Again, the difficulty is in the analysis of the behaviour Markov Chains. Further details are omitted.

## 6 LogSpace Machines as Statistical Tests

By a logspace machine we mean a machine which have online access to its input and work in space logarithmic in the length of its input.

### 6.1 Characterization of LogSpace Statistical Tests

In light of the fair amount of understanding gained in the study of logspace pseudorandomness, it is not outrageous to propose the following open problem.

**Open Problem 2** : Try to find a useful characterization of logspace statistical tests.

By a useful characterization, we mean one which would lead to a "simple" and "natural" universal test for the class of logspace statistical tests which accept all but a *specified* fraction of the strings. We do not consider the universal test of the proceeding subsection as being "natural" or "simple".

### 6.2 Universal Tests for Subclasses of LogSpace Tests

Following is a description of a universal test for the subclass of logspace statistical tests which accept all but a *specified* negligible fraction of the strings. Unlike the tests presented in previous sections, the current universal test does not have a natural description, but rather consists of executing all relevant tests. In other words, the universal test presented below uses the "diagonalization paradigm". Namely, it enumerates all Turing Machines (up to a bound depending on the input length), and simulating the execution of each machine on the given input with a specific space-bound. The space bound is a function which also depends on the input length and has the property that it is bigger than the logarithm of the input length (to any base). The verdict of a simulated test on the given input is taken into consideration (as grounds for rejecting) only if the test is found to accept all but a *specified* negligible fraction of the strings of the given length (i.e., the length of the given input). Using the results of Nisan [15, 16], it is possible to implement the above universal test in slightly more than polynomial-time and in polylogarithmic space. Namely,

**Proposition 4** *Let $f : \mathbb{N} \mapsto \mathbb{N}$ be any polynomial-time computable function which is unbounded and monotonically non-decreasing (e.g., $f(n) = \log^* n$). Then there exists a deterministic universal test for the class of logspace statistical tests which accept all but a $\frac{1}{n^{f(n)}}$ fraction of the strings of length $n$. Furthermore, on input of length $n$, the universal test runs for at most $O(n^{f(n)})$ steps and uses at most $O(f(n) \log n)^2$ space.*

## 6.3 Pseudorandomness and Acceptance by Statistical Tests

By Proposition 1, all ensembles dominated by (logspace)-pseudorandom ensembles are accepted by all logspace statistical tests. We now show that these are not all the ensembles which are accepted by logspace tests. Namely,

**Proposition 5** *Let $l(n) \stackrel{\text{def}}{=} \log_2 n$, and $t(n) \stackrel{\text{def}}{=} 2n + 4 + 2\log_2 n$. Let $X = \{X_{t(n)}\}_{n \in \mathbb{N}}$ be an ensemble so that for each $\alpha \in \{0,1\}^n$ we have $\text{Prob}(X_n = 01^{l(n)}0\alpha01^{l(n)}0\alpha) = 2^{-n}$. Then the ensemble $X$ is accepted by all logspace statistical tests and yet is not dominated by any logspace-pseudorandom ensemble.*

**proof sketch:** We first show that the ensemble $X$ is indeed accepted by all logspace statistical tests. Let $T$ be an arbitrary logspace statistical test. To show that $T$ must accept $X$, we use a standard "crossing sequence" argument. Denote by $P_n(\gamma)$ the set of $n$-bit strings $\alpha$ such that after reading the prefix $01^{l(n)}0\alpha$ (of the input) machine $T$ is in configuration $\gamma$. Likewise, denote by $S_n(\gamma)$ the set of $n$-bit long strings $\beta$ such that starting from configuration $\gamma$ and reading the input suffix $01^{l(n)}0\beta$ the machine $T$ rejects. Now, assume to the contradiction that $T$ does not accept $X$. Then there exists a $c > 0$ such that for infinitely many $n$'s $\text{Prob}(T(X_{t(n)}) = \text{REJ}) > \frac{1}{n^c}$. It follows that for these $n$'s

$$|\{\alpha : \exists\gamma \text{ s.t } \alpha \in P_n(\gamma) \text{ and } \alpha \in S_n(\gamma)\}| > \frac{2^n}{n^c}$$

Since on inputs of length $t(n)$ the algorithm $T$ can assume only polynomially (in $n$) many configurations, there must exists a sequence of $\gamma_n$'s such that for some $d > 0$ and for infinitely many $n$'s

$$|\{\alpha : \alpha \in P_n(\gamma_n) \text{ and } \alpha \in S_n(\gamma_n)\}| > \frac{2^n}{n^d}$$

However, it follows that every input of the form $01^{l(n)}0\alpha 01^{l(n)}0\beta$, with $\alpha \in P_n(\gamma_n)$ and $\beta \in P_n(\gamma_n)$, is rejected by $T$. Since the fraction of these inputs (among all $t(n)$-bit long inputs) is at least $(2^{-(l(n)+2)} \cdot \frac{1}{n^d})^2 = \frac{1}{16 \cdot n^{2d+2}}$, we derive a contradiction to the hypothesis that $T$ is a statistical test.

We now show that $X$ cannot be dominated by any pseudorandom ensemble. Assume to the contrary that $X$ is dominated by $Y$ and $Y$ is logspace pseudorandom. Even under the weak definition of dominance (i.e., Definition 8) the probability mass assigned by $Y_{t(n)}$ to strings of the form $01^{l(n)}0\alpha 01^{l(n)}0\alpha$, where $\alpha$ is an $n$-bit long string which does not contain the substring $1^{l(n)}$, cannot be negligible (since such strings are assigned constant probability mass by $X$ and hence cannot be assigned negligible probability by $Y$). Hence for some $c > 0$ and infinitely many $n$'s $\text{Prob}(Y_{t(n)} \in S_{t(n)}) > \frac{1}{n^c}$, where

$$S_{t(n)} \stackrel{\text{def}}{=} \{01^{l(n)}0\alpha 01^{l(n)}0\alpha \, : \, \alpha \in G_n\} \, ,$$

where $G_n \subset \{0,1\}^n$ is the set of all $n$-bit long strings which do not contain the substring $1^{l(n)}$. Using this fact we present a logspace algorithm, $A$, (which is *not* a statistical test) that distinguishes the ensemble $Y$ from the uniform ensemble. Algorithm $A$ tries to compare $2c\log_2 n$ positions in the prefix of the input with the corresponding positions in the suffix, and outputs 1 if and only if the bits in these positions match. Following is an oversimplified implementation (of this idea) which does not work: on input $\sigma_1 \cdots \sigma_{t(n)}$ ($\sigma_i \in \{0,1\}$), the algorithm outputs 1 if and only if for all $j \in \{1, ..., 2c\log_2 n\}$ we have $\sigma_j = \sigma_{\frac{t(n)}{2}+j}$. The problem with this oversimplified algorithm is that it cannot be implemented in logspace. Yet the algorithm has the desirable property of distinguishing $Y$ from the uniform ensemble (since on input taken from $Y_{t(n)}$ the algorithm outputs 1 with probability at least $\frac{1}{n^c}$, whereas on a uniformly chosen input the algorithm outputs 1 with probability exactly $\frac{1}{n^{2c}}$).

We now present a modified implementation of the above idea. Let $p_1, p_2, ...$ be a binary sequence such that $p_i = 1$ if and only if there exists an integer $j$ so that $i = \lfloor 2^{\frac{j}{2c}} \rfloor$. Clearly $p_i$ can be determined in logarithmic space in $i$. Furthermore, for every $m$ the cardinality of the set $\{i : i \leq m \bigwedge p_i = 1\}$ is at most $2c\log_2 m$ and at least $(2c\log_2 m) - 4c$. On input $\sigma_1 \cdots \sigma_{t(n)}$ ($\sigma_i \in \{0,1\}$), algorithm $A$ operates in three phases. In the first phase, it scans the input until finding a prefix of the form $01^L0$. The algorithm records $L$ (in binary). In the second phase the algorithm scans the next substring of the input and records the bits in the positions for which $p_i = 1$ (i.e., bit $\sigma_{L+2+i}$ is recorded iff $p_i = 1$). The second phase is terminate once another $01^L0$ substring is found. The algorithm records (again in binary) the number of bits, $M$, scanned in the second phase. In the third phase the algorithm scans the rest of the input and checks whether the bits in this substring, with positions having $p_i = 1$, equal to the corresponding recorded bits. Namely, for $p_i = 1$, the algorithm checks whether $\sigma_{L+2+i}$ (recorded in the second phase) equals $\sigma_{L+2+M+i}$ (currently being scanned). If one of these checks yields a negative result, the algorithm halt immediately outputting 0. In case the third stage is completed successfully, the algorithm checks whether the number of bits scanned in the third phase equals $N \stackrel{\text{def}}{=} M - (L+2)$, and whether $N = 2^L$. If either checks yields a negative result then the algorithm halts outputting 0 else the output is 1.

One can easily verify that the above algorithm $A$ can be implemented in logarithmic space (i.e. it uses space $(c+O(1))\cdot\log_2 n$ on inputs of length $n$). Clearly algorithm $A$ outputs 1 on every input in $S_{t(n)}$. Using $\mathrm{Prob}(Y_{t(n)}\in S_{t(n)})\geq\frac{1}{n^c}$ it follows that $\mathrm{Prob}(A(Y_{t(n)})=1)\geq\frac{1}{n^c}$. On the other hand, on input $\sigma_1\cdots\sigma_{t(n)}$, the algorithm outputs 1 only if $\sigma_{l(n)+2+i}=\sigma_{2l(n)+4+n+i}$ for every $i\leq n$ satisfying $p_i=1$ (and there are at least $2c\log_2 m-4c$ such $i$'s). Hence, for the uniform distribution $U_{t(n)}$ we have $\mathrm{Prob}(A(U_{t(n)})=1)<2^{4c-2c\log_2 n}<\frac{16^c}{n^{2c}}$. The proposition follows. $\square$

**Open Problem 3** : Suppose that the ensemble $X$ passes all logspace statistical tests. What can be said about $X$ in terms of pseudorandomness with respect to logspace distinguishers.

## 7 Probabilistic Polynomial-Time Statistical Tests

It would be more natural to consider *deterministic* polynomial-time statistical tests, rather than *probabilistic* polynomial-time ones. Unfortunately, we have results only for probabilistic polynomial-time statistical tests. Our main result relates pseudorandom generators and "generators" which generate ensembles which are accepted by all probabilistic polynomial-time statistical tests. We first recall the definition of a pseudorandom generator.

**Definition 9** (pseudorandom generator [1, 19]): *A (deterministic) polynomial-time algorithm $G$ is called a* pseudorandom generator (prg) *if the following two conditions hold:*

1. *$|G(\alpha)|\geq|\alpha|+1$, for all $\alpha\in\{0,1\}^*$.*
2. *The ensemble $\{G_n\}_{n\in\mathbb{N}}$ defined by letting $G_n=G(U_n)$ is pseudorandom (with respect to probabilistic polynomial-time algorithms).*

It is well known that if pseudorandom generators exist then for, every polynomial $p$, there exists pseudorandom generators $G$ satisfying $|G(\alpha)|=p(|\alpha|)$, for all $\alpha\in\{0,1\}^*$. By a sequence of results, culminating in [7], pseudorandom generators exist if and only if one-way functions exist. Interestingly, this condition is also equivalent to the existence of *statistically-accepted generators* defined below.

**Definition 10** (statistically-accepted generator): *A (deterministic) polynomial-time algorithm $G$ is called a* statistically-accepted generator *if the following two conditions hold:*

1. *$|G(\alpha)|\geq 2\cdot|\alpha|$, for all $\alpha\in\{0,1\}^*$.*
2. *The ensemble $\{G_n\}_{n\in\mathbb{N}}$ defined by letting $G_n=G(U_n)$ is accepted by all probabilistic polynomial-time statistical tests.*

**Theorem 7** *Pseudorandom generators exist if and only if statistically-accepted generator exist.*

**Proof:** Clearly, every pseudorandom generator with sufficient expansion (i.e., with $|G(\alpha)| \geq 2|\alpha|$ for all $\alpha$) is a statistically-accepted generator. The theorem follows by proving that the existence of a statistically-accepted generator implies the existence of a one-way function (and hence the existence of pseudorandom generators). To this end, we note that the construction used by Levin [13], to prove that the existence of pseudorandom generators implies the existence of one-way functions, proves in fact the stronger statement we just made. For sake of self-containment, this construction is repeated below.

Let $G$ be an arbitrary statistically-accepted generator. Without loss of generality, assume that $|G(\alpha)| = 2|\alpha|$, for all $\alpha \in \{0,1\}^*$. Define the function $f$ so that $f(\alpha\beta) \stackrel{\text{def}}{=} G(\alpha)$, where $|\alpha| = |\beta|$. We now claim that $f$ is a one-way function. Assume on the contrary, that a probabilistic polynomial-time algorithm, $A$, can invert $f$ with probability that is not negligible[7]. Hence, on a fraction which is not negligible of the strings, $s$, the algorithm $A$ on input $f(s)$ outputs with non-negligible probability a string $y$ so that $f(s) = f(y)$. Algorithm $A$ can be easily modified so that on a fraction which is not negligible of the strings, $A$ can invert successfully with probability at least $\frac{2}{3}$. Consider the following algorithm $T$. On input $x$, algorithm $T$ first computes $y \stackrel{\text{def}}{=} A(x)$, next checks whether $f(y) = x$, and finally outputs 1 if and only if the answer is negative. Clearly, $T$ constitutes a probabilistic polynomial-time statistical test (since all but a negligible fraction of the strings do not have an inverse under $f$). However, the probability that $T$ rejects a string taken from $G_n$, is not negligible. Hence, $T$ does not accept the ensemble generated by $G$, in contradiction to the hypothesis that $G$ is a statistically-accepted generator. $\Box$

**Remark 7** : The above theorem holds also if a statistically-accepted generator is only required to expand strings of length $n$ into strings of length $n + l(n)$, where $l$ is an arbitrary function growing faster than any logarithm in $n$. However, expansion of strings of length $n$ into strings of length $n + O(\log n)$ does not suffice (since, for example, $G(\alpha) = \alpha 10^{\log_2 |\alpha|}$ generates an ensemble accepted by all probabilistic polynomial-time statistical tests and yet this does not imply the existence of one-way functions)!

**Remark 8** : Clearly, if statistically-accepted generators exist then for, every polynomial $p$, there exists statistically-accepted generators $G$ satisfying $|G(\alpha)| = p(|\alpha|)$, for all $\alpha \in \{0,1\}^*$.

**Remark 9** : We stress that the above theorem sheds no light on the open problem mentioned in Section 3. Namely, are all ensembles accepted by (probabilistic) polynomial-time tests also dominated by pseudorandom ensembles?

## Acknowledgements

[7]Recall, that a function $f$ is one-way if all probabilistic polynomial-time algorithms invert it with only negligible probability. Namely, for every such algorithm $A$, the sequence of probabilities $\{p_n\}_{n \in \mathbb{N}}$ is negligible, where $p_n \stackrel{\text{def}}{=} \text{Prob}(A(f(X)) \in f^{-1}f(X) | X = U_n)$.

We would like to thank Jon Aaronson, Noga Alon, Amir Dembo, Nati Linial, and Ofer Zeitouni, for indispensable discussions and references to Large Deviation and Local Limit Theorems for finite Markov Chains. We are also grateful to Ishai Ben-Aroya for pointing out an error in an early version of this extended abstract.

## References

[1] Blum, M., and Micali, S., "How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits", *SIAM Jour. on Computing*, Vol. 13, 1984, pp. 850-864. Extended abstract in *FOCS 82*.

[2] Chaitin, G., "A Theory of Program Size Formally Identical to Information Theory", IBM Yorktown Heights Rep. RC 4805, April 1974.

[3] Dembo, A., and O. Zeitouni, *Large Deviation and Applications*, preprint, 1991.

[4] Fredricksen, H.M., "Generation of the Ford sequence of length $2^n$, $n$ large", *Journal of Combinatorial Theory*, Series A, Vol. 12 (1972), pp. 153-154.

[5] Goldwasser, S., and S. Micali, "Probabilistic Encryption", *JCSS*, Vol. 28, No. 2, 1984, pp. 270-299. Extended abstract in *STOC 82*.

[6] Gnedenko, B.V., *The Theory of Probability*, (translated from fourth Russian edition by B.D. Seckler), Chelsea Publishing Company, New York (NY), 1968.

[7] Hastad, J., R. Impagliazzo, L.A. Levin, and M.G. Luby, "Pseudorandom Generators from any One-Way Functions", preprint. Combines a work of Impagliazzo, Levin and Luby in *21st STOC* (1989) with a work of Hastad in *22nd STOC* (1990).

[8] Kemeny, J.G., and J.L. Snell, *Finite Markov Chains*, D. Van Nostrad Company, Inc., Princeton (NJ), 1960.

[9] Knuth, D.E., *The Art of Computer Programming*, Vol. 2 (*Seminumerical Algorithms*), Addison-Wesley Publishing Company, Inc., 1969 (first edition) and 1981 (second edition).

[10] Kolmogorov, A.N., "Three Approaches to the Quantitative Definition of Information", *Probl. Inform. Transmission*, Vol. 1, pp. 1-7, 1965.

[11] Lempel, A., and J. Ziv , "On the Complexity of Finite Sequences", *IEEE Trans. on Information Theory*, Vol. IT-22, No. 1, Jan. 1976, pp. 75-81.

[12] L.A. Levin, "Average Case Complete Problems", *SIAM J. of Computing*, Vol. 15, 1986, pp. 285-286. Extended abstract in *STOC 84*.

[13] L.A. Levin, "One-way Functions and Pseudorandom Generators", *Combinatorica*, Vol. 7, No. 4, 1987, pp. 357–363.

[14] Martin-Löf, P., "The Definition of Random Sequences", *Inform. Contr.*, Vol. 9, pp. 602-619, 1966.

[15] N. Nisan, "Pseudorandom Generators for Space Bounded Machines", *22nd STOC*, 1990.

[16] N. Nisan, "RL $\subseteq$ SC", *23rd STOC*, 1991.

[17] Rousseau-Egele, W., "Un Theoreme de la Limite Locale pour une Classe de Transformations Dilatantes et Monotones par Morceaux", *Ann. Prob.*, 11 (1983), pp. 772-788.

[18] Solomonoff, R.J., "A Formal Theory of Inductive Inference", part 1 and 2, *Information and Control*, Vol. 7, 1964, pp. 1-22 ans 224-254.

[19] Yao, A.C., "Theory and Applications of Trapdoor Functions", *Proc. of the 23rd IEEE Symp. on Foundation of Computer Science*, 1982, pp. 80-91.

[20] Ziv. J., and A. Lempel, "A Universal Algorithm for Sequential Data Compression", *IEEE Trans. on Information Theory*, Vol. IT-23, No. 3, May 1977, pp. 337-343.

[21] Ziv, J., and N. Merhav, "A Measure of Relative Entropy between Individual Sequences with Applications to Universal Classification", preprint.