Random Polynomial Time is Equal to Semi-Random Polynomial Time

Umesh V. Vazirani*
Vijay V. Vazirani**

88-959

December 1988

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

*Research supported by NSF grant DCR 85-13926 and a PYI Award.
**Research supported by an NSF PYI Award with matching funds from AT&T Bell Labs and Sun Microsystems, Inc.

Random Polynomial Time is Equal to

Semi-random Polynomial Time.

Umesh V. Vazirani*
Computer Science Department
University of California, Berkeley

Vijay V. Vazirani**
Computer Science Department
Cornell University

ABSTRACT

We prove that any one-sided error random polynomial time (RP) algorithm can be simulated with a semi-random source at no more than polynomial factor loss in efficiency. i.e. RP = SRP. This contrasts with the fact that a semi-random source is too weak to simulate fair coin flips [SV].

* Supported by NSF Grants DCR 85-13926 and a PYI Award. ** Supported by a PYI Award with matching funds from AT&T Bell Labs and Sun Microsystems, Inc.

1. Introduction

The successful algorithmic use of randomness for solving some basic computational problems has led to serious consideration of randomness as a computational resource. Physical sources of randomness suffer from problems of correlation and therefore cannot provide the kind of perfect randomness assumed in the above algorithms. This deficiency can be expected to become more acute as algorithms make increasingly sophisticated use of randomness, and their correctness proofs rely more critically on the independence of the random bits. It is therefore necessary to develop an adequate theory for utilizing a physical source of randomness for efficient computation. Following up on earlier work in this direction [Bl], a mathematical model for such sources - the semi-random source - was proposed in [SV]. The simplicity and modest randomness assumptions of this model make it a viable starting point for such a theory. In this paper we show that any one-sided error random polynomial time (RP) algorithm can be efficiently simulated (i.e. with no more than polynomial factor loss in efficiency) with a semi-random source, thereby proving that RP = SRP.

In the past, one direction of research has explored whether it is possible to dispense with randomization altogether at no more than a polynomial factor increase in the running time; i.e. 'Is P = RP?'. Some progress in this direction has been made by exploiting the fact that polynomial time algorithms cannot distinguish the difference if they are fed sufficiently high quality pseudo-random strings in place of random strings. Under a certain hypothesis about the existence of one-way functions, such pseudo-random sequences can be obtained from shorter random strings [BM, Ya], and yield a sub-exponential simulation of randomized polynomial time computations [Ya].

Our approach differs from the above in two ways: firstly we assume that the witness set (i.e. for a given input, those choices for the random string on which the randomized algorithm gives the correct answer) is an arbitrary set containing half the n-bit strings; this allows us to obtain a result that is independent of any unproven complexity-theoretic assumptions. Secondly, we exploit the fact that once the randomized algorithm and the input string are fixed, so is the witness set. Since the algorithm makes only one-sided errors, it can be simulated efficiently by picking an element of the fixed witness set in polynomially many trials. This requirement is formulated via the notion of the spread of a probability distribution on a collection of sets of n-bit strings. Informally, a probability distribution over the collection of k-sized sets of n-bit strings has a large *spread* if for every choice, W, of the witness set, with high probability a set drawn from this distribution does not get trapped in $\{0,1\}^n - W$. Our main theorem can now be stated as follows: we show how to efficiently convert a semi-random probability distribution into one having a large spread; k will be bounded by a polynomial in n. Since a semi-random distribution cannot be converted into a truly-random one [SV], this shows that the task of obtaining a distribution having a large spread is easier than that of obtaining a truly-random one.

The notion of spread can be generalized appropriately for algorithms with two-sided error. In [Va4], a modification of the transformation given here is shown to produce a distribution with large (two-sided) spread from any semi-random distribution. This shows that any two-sided-error probabilistic polynomial time (BPP) algorithm can be simulated in polynomial time using a single semi-random source. The solution to this more general problem has an interesting consequence - the basic transformation can be bootstrapped to achieve a more efficient simulation, in particular also for RP. In [CG], it was shown that this transformation works even under more severe assumptions on the distribution produced by the source of randomness. In a different direction, assuming that the witness set is a random set of n-bit strings, [AR] show that substituting semi-random strings in place of truly-random strings in the original randomized algorithm still yields a witness with constant probability.

[Sa] considers the following question: how many truly random bits are required to produce a distribution (on polynomially longer strings) with large spread. Via a counting argument [Sa] showed that there exist transformations that can convert a truly random distribution on a short string into a distribution with large spread. If efficient transformations satisfying this property were found, then the failure probability of any (one-sided error) randomized algorithm can be made exponentially small without increasing the number of random bits used. It has been conjectured [Va2] [Va3] that practical pseudo-random number generators are effective precisely because they achieve such a transformation. Two good candidates for this thesis are the linear congruential generators and shift register sequence generators. In each case it has been repeatedly observed that randomized algorithms work very well in practice on the output of these generators. However, these generators are known to be efficiently distinguishable from truly-random sources (see for example [FKL] and [Pl]), and therefore are not perfect in the sense of [Ya]. A new pseudo-random number generator based on random walks on expander graphs is given in [AKS]; the output of this generator has provably large spread.

2. Preliminaries

Definition [SV]: A distribution on an infinite sequence of 0/1 random variables $x_1, x_2, \cdots$ is δ *semi−random* if for every positive integer i and every 0/1 string s of length $i - 1$:

$$\delta \le Pr\ [x_i = 1 \mid x_1 \cdots x_{i-1} = s\,] \le 1 - \delta$$

where $0 < \delta \le \frac{1}{2}$, and by the event '$x_1 \cdots x_{i-1} = s$' we mean the event that x_1 is the first bit of s, x_2 the second bit and so on. A δ *semi-random source* is a source whose output distribution is δ semi-random. A *semi-random source* is a δ semi-random source, for some constant $\delta, 0 < \delta \le \frac{1}{2}$.

In order to evaluate the worst-case performance of the algorithm that utilizes a semi-random source, it is convenient to regard the semi-random source as a process controlled by an *adversary* which generates a sequence of 0's and 1's by flipping a coin of variable bias. The adversary has infinite computing power, and complete knowledge of the algorithm and the bit-sequence generated so far. The adversary sets the bias of the next bit to attenuate the performance of the algorithm. The only restriction is that the bias must be in the range $[\delta,\ 1-\delta]$. The map T: $\{0,1\}^{<\omega} \to [\delta, 1-\delta]$, where for $s \in \{0,1\}^{<\omega}$, $T(s) = Pr[x_{|s|+1} = 1 \mid x_1 \cdots x_{|s|} = s]$ will be called the *strategy* of the adversary.

The following theorem shows that a semi-random source cannot be used to simulate a fair coin:

Theorem [SV]: Let $f : \{0,1\}^m \to \{0,1\}$ denote an arbitrary boolean function. Then there is an adversary strategy, such that $f(x)$ is at least $1 - \delta$ biased when x is generated by a semi-random source with parameter δ.

In contrast, the following theorem states that two independent semi-random sources can be used to arbitrarily closely simulate a fair coin:

Theorem [Va1]: There is an algorithm that converts $O(\frac{1}{\delta^2}n)$ bits from two independent semi-random sources into an n-bit quasi-random (i.e. almost uniformly generated; see [Va] for exact definition) sequence.

Definition [Gi]: A language L is in *Random Polynomial Time (RP)* if there exists polynomial p, and a deterministic polynomial time two-input algorithm M:

(i) if $x \in L, M[x,r]$ accepts for at least half the strings, r, of length $p(|x|)$.

(ii) if $x \notin L, M[x,r]$ rejects for each string r of length $p(|x|)$

Let $p(|x|) = n$ and $W(x) = \{r \in \{0,1\}^n \mid M[x,r]\}$ accepts. Clearly, if $x \in L$, $|W(x)| \geq 2^{n-1}$. $W(x)$ is called the *witness set* and its elements are called *witnesses.*

Definition: Semi-random Polynomial Time (SRP) is defined analogously. Let δ be a constant, $0 < \delta \leq \frac{1}{2}$. A language L is in δSRP (δ *Semi–random Polynomial Time*) if there exists a polynomial p and a polynomial time two input algorithm M:

(i). if $x \in L, M[x,r]$ accepts with probability at least $\frac{1}{2}$

where r is a string of length $p(|x|)$ output by an arbitrary δ semi-random source.

(ii). if $x \notin L, M[x,r]$ rejects for each string r of length

$p(|x|)$.

(Notice that (i) must hold for *every* δ semi-random source).

Now, $SRP = \bigcap_{0 < \delta \leq \frac{1}{2}} \delta SRP.$

3. The Spread of a Distribution

Since at present no special properties of witness sets of RP languages are known, we will assume that the witness set is an *arbitrary* subset of $\{0,1\}^n$ of cardinality 2^{n-1}. Now, proving RP=SRP is essentially equivalent to exhibiting a polynomial time algorithm that uses a δ semi-random source to generate polynomially many queries such that with probability at least 1/2, at least one of the query strings is guaranteed to be a witness. The required properties of the induced probability distribution on the query strings are captured in the notion of spread:

Definition: Let $U_{n,k}$ denote the collection of all multisets of cardinality k whose elements are n-bit sequences, and let D be a probability distribution on $U_{n,k}$. Let S denote a set picked from $U_{n,k}$ according to D. Define *spread* $(D) = \min_{W \subseteq \{0,1\}^n,\ |W| = 2^{n-1}} \{Pr_D\ [S \cap W \neq \emptyset]\}$.

Informally, a distribution with small spread can get 'trapped' in a suitably chosen non-witness set. Notice that a δ semi-random distribution can have an inverse-exponentially small spread. The algorithm mentioned above is required to transform each δ semi-random distribution into a distribution having an inverse polynomial spread.

4. Transforming Semi-random Distributions into Distributions with Large Spread

Consider the transformation from k n-bit strings to 2^k n-bit strings $t: U_{n,k} \rightarrow U_{n,2^k}$ such that for $S = \{s_1 \cdots s_k\}$, $t(S) = \{\bigoplus_{s_i \in X} s_i : X \subseteq S\}$, where $\oplus$ denotes bit-wise GF[2] sum.

Theorem 1: Let $k = \left\lceil \frac{1}{\delta} \log n \right\rceil$, and let $S = \{s_1, \ldots, s_k\}$ be a set of k successive n-bit strings output by a δ semi-random source. Let D be the distribution on $U_{n,2^k}$ induced by the transformation t. Then for every δ and every δ semi-random distribution, spread (D) is at least inverse polynomial in n.

Remark: For the parameters stated in Theorem 1, $t(S)$ can clearly be computed in time polynomial in n.

Bijections Induced by Strings:

We first lay the groundwork for the proof of Theorem 1. Associate with each string $s \in \{0,1\}^n$ a bijection $h_s : \{0,1\}^n \rightarrow \{0,1\}^n$ given by:

$$h_s(x) = s \oplus x.$$

Let $B_{n,l}$ denote the collection of all multisets of cardinality l whose elements are bijections on $\{0,1\}^n$, and let $H \in B_{n,2^k}$ be the set of bijections associated with the strings in $t(S)$. To establish large spread of distribution D, it will suffice to show that the associated distribution on bijections satisfies the condition described below.

Fix an arbitrary witness at $W \subseteq \{0,1\}^n$, $|W| = 2^{n-1}$.

Definition: The *effective non−witness set* *w.r.t.* bijections $\rho_1 \cdots \rho_m$ on $\{0,1\}^n$ is:

$$\{ x \in \{0,1\}^n \mid \rho_i(x) \notin W \text{ for } 1 \le i \le m \}$$

The significance of the above definition lies in the fact that evaluating the bijections at any point outside the effective non-witness set yields at least one witness. We shall show that for every choice of W, the effective non-witness set with respect to the set H of bijections is ϕ with at least inverse polynomial probability. This will suffice to prove theorem 1, since the strings in $t(S)$ are simply the evaluation of the bijections in H at 0^n.

In order to prove the above statement, the following structure of H, inherited from $t(S)$, will be useful: Let $\{g_1, \ldots, g_k\} \in B_{n,k}$ be the set of bijections associated with S, $H_0 = \{I\}$, and $H_{i+1} = H_i \cup \{h \circ g_{i+1} \mid h \in H_i\}$, for $0 \le i < k$. Then $H = H_k$.

Now let A_i denote the effective non-witness set *w.r.t.* the bijections in H_i, for $1 \le i \le k$, and let $A_0 = \{0,1\}^n - W$. To prove Theorem 1 it will suffice to show that:

$\forall W \; Pr_D \, [A_l = \varnothing]$ is at least inverse polynomial in n.

First we give a recursive characterization of A_i:

Lemma 1: Under the above definitions,

$$A_{i+1} = A_i \cap g_{i+1}(A_i), \; 0 \le i < l.$$

Proof: Using the recursive definition of H_i:

$$\begin{aligned} A_{i+1} &= \{ x \in A_i \mid \forall g \in H_i, \; g \circ g_{i+1}(x) \notin W \} \\ &= \{ x \in A_i \mid g_{i+1}(x) \in A_i \} \\ &= A_i \cap g_{i+1}(A_i) \qquad \square \end{aligned}$$

As a consequence of Lemma 1, proving that D has a large spread is reduced to analyzing the action of a "typical" bijection on an arbitrary set A. A vanishingly small fraction of all bijections will leave A invariant (i.e. map A onto A). We must prove a stronger statement: for bijections of the form $g_s(x) = s \oplus x$ where s is picked from a semi-random distribution, the typical bijection "moves" A significantly, i.e. $E(|A \cap g_s(A)|)$ is much smaller than A.

Say that an adversary strategy is *extreme* if it is a function $f: \{0,1\}^{<w} \rightarrow \{\delta, 1-\delta\}$, i.e. the biases are either δ or $1-\delta$, and no intermediate values. By the following lemma, it is sufficient to consider extreme strategies only.

Lemma 2: Let $A \subseteq \{0,1\}^n$ and let $A' = (A \cap g_s(A))$, where s is a δ semi-random sequence. Then there is an extreme strategy which maximizes $E(|A'|)$.

Proof: Let T be an arbitrary strategy. It is easy to see that T can be expressed as a convex combination of extreme strategies, i.e. for each string $s \in \{0,1\}^n$, the probability of generating s under T is a certain convex combination of the probabilities of generating s under the extreme strategies. Now $E(|A'|)$ under T will be this same convex combination of the expectations of $(|A'|)$ under the extreme strategies. Since a convex combination is maximized at an extreme point, the lemma follows.□

Notation: If $A \subseteq \{0,1\}^n$, then the fraction $\frac{|A|}{2^n}$ will be denoted by $\mu(A)$. Let s be an n-bit δ semi-random string, and $A, B \subseteq \{0,1\}^n$, then $\frac{|A \cap g_s(B)|}{2^n}$ will be denoted by $\mu_s(A,B)$. For $x \in \{0,1\}^n$, $0x$ will denote the string 0 concatenated with x. Finally let a be the constant $\frac{1}{2(1-\delta)}$.

Lemma 3: Let $A \subseteq \{0,1\}^n$, and let s be an n-bit δ-semi-random string. Then $E(\mu_s(A,A)) \leq (\mu(A))^{\frac{1}{1-\delta}}$.

Proof: By an induction on n, we will prove the following stronger statement: let $A, B \subseteq \{0,1\}^n$, then

$$E(\mu_s(A,B)) \leq \mu(A)^a \mu(B)^a.$$

The statement is easily verified for $n = 1$. Assume it is true for n. Let $A, B \subseteq \{0,1\}^{n+1}$.

Let $A_0 = \{x \in \{0,1\}^n \mid 0x \in A\}$ and $A_1 = \{x \in \{0,1\}^n \mid 1x \in A\}$. Define B_0 and B_1 analogously. Clearly, $\mu(A_0) + \mu(A_1) = 2\mu(A)$ and $\mu(B_0) + \mu(B_1) = 2\mu(B)$.

By lemma 2 it is sufficient to consider only extreme strategies. Moreover, *w.l.o.g.*, we may assume that the first bit of s, i.e. s_1, is 0 with probability $1-\delta$. Let w denote the last $n-1$ bits of s.

$$\therefore E(\mu_s(A,B)) = E(\mu_s(A,B) \mid s_1 = 0)(1-\delta)$$

$$+ E(\mu_s(A,B) \mid s_1 = 1)(\delta)$$

$$= \tfrac{1}{2} E(\mu_w(A_0, B_0) + \mu_w(A_1, B_1))(1-\delta) + \tfrac{1}{2} E(\mu_w(A_0, B_1) + \mu_w(A_1, B_0))\,\delta$$

$$\le \tfrac{1}{2}[\mu(A_0)^a\,\mu(B_0)^a + \mu(A_1)^a\,\mu(B_1)^a](1-\delta) + \tfrac{1}{2}[\mu(A_0)^a\,\mu(B_1)^a + \mu(A_1)^a\mu(B_0)^a]\,\delta$$

$$= \tfrac{1}{2}[\mu(A_0)^a\,\mu(B_0)^a + (2\,\mu(A) - \mu(A_0))^a\,(2\,\mu(B) - \mu(B_0))^a\,](1-\delta) +$$

$$\tfrac{1}{2}[\,\mu(A_0)^a\,(2\,\mu(B) - \mu(B_0))^a + (2\,\mu(A) - \mu(A_0))^a\,\mu(B_0)^a\,]\,\delta$$

Substituting $\dfrac{\mu(A_0)}{\mu(A)} = x$ and $\dfrac{\mu(B_0)}{\mu(B)} = y$,

$$E(\mu_s(A,B)) \le \tfrac{1}{2}\,\mu(A)^a\,\mu(B)^a\,\{(1-\delta)[x^a y^a + (2-x)^a (2-y)^a] + \delta[x^a(2-y)^a + (2-x)^a y^a]\}.$$

Via an elementary, though tedious, computation one can show that the function $f(x,y) = (1-\delta)\,[x^a\, y_a + (2-x)^a\,(2-y)^a\,] + \delta\,[x^a\,(2-y)^a + (2-x)^a\,y^a]$ attains its maximum at $x=y=1$, under the following conditions: $0 \le x,\ y \le 2,\ a = \dfrac{1}{2(1-\delta)},\ 0 < \delta \le \tfrac{1}{2}$ (for a detailed proof see [VV1]); the maximum value being 2.

$$\therefore E[\mu_s(A,B)] \le \mu(A)^a\,\mu(B)^a.$$

This proves the lemma. □.

Proof of Theorem 1: Let A_i, $0 \le i \le l$, be the sets defined above w.r.t. a fixed witness set $W \subseteq \{0,1\}^n$, $|W| = 2^{n-1}$. As a consequence of lemma 3,

$$P_r\,[\mu(A_{i+1}) < \mu(A_i)^{1+\delta}] \le 1 - (\mu(A_i))^{\frac{\delta^2}{1-\delta}}$$

$$\le 1 - \left(\frac{1}{2}\right)^{\frac{\delta^2}{1-\delta}}$$

since $\mu(A_i) \le \dfrac{1}{2}$.

If $\mu(A_i)$ drops by an exponent of $(1+\delta)$ at each step,

$$\mu(A_k) < \mu\,(A_0)^{(1+\delta)^k}$$
$$= (\frac{1}{2})^{(1+\delta)^k}$$
$$< \frac{1}{2^n}$$

$\Rightarrow A_k = \phi.$

The probability that $\mu(A_i)$ drops by an exponent of $(1+\delta)$ at each step is lower bounded by $(1-(\frac{1}{2})^{\frac{\delta^2}{1-\delta}})^k$, which is inverse polynomial in n. Taking the product of the success probability at each step is justified by the following property of a δ semi-random distribution: let $x_1, x_2, \cdots$ be $0/1$ random variables satisfying the δ semi-random condition, then $x_i, x_{i+1}, \cdots$ have a δ semi-random distribution even if they are conditioned on the event $x_1, x_2, \cdots x_{i-1} = s$, for any string $s \in \{0,1\}^{i-1}$.

Since W is an arbitrary witness set, the theorem follows. □

Corollary: RP = SRP.

Proof: By a polynomial number of repetitions of the above scheme, the probability of error can be reduced to less than 1/2. This is so because Theorem 1 gives a uniform upper bound on the probability of failure of a run of the algorithm. As in the proof of Theorem 1, by the semi-randomness property, the product of these failure probabilities is an upper bound on the failure probability of the entire scheme. The overall running time is polynomial in n, though δ occurs in the exponent. □

5. Discussion

We have introduced a measure for evaluating the distribution produced by a source of randomness for the purpose of simulating randomized algorithms. This measure, the spread of the distribution, evaluates the probability of picking a witness for the worst choice of the witness set.

In this paper, we show that distributions with large spread are easier to produce than truly random distributions: semi-random distributions can be transformed into the former, but not the latter. In the same spirit, a definition of two-sided spread is given in [Va4] to evaluate the worst-case performance of a probability distribution over all two-sided error randomized algorithms. In that paper, it is shown how to efficiently transform any semi-random distribution into a distribution with large two-sided spread. An advantage of this stronger result is that the transformation can be bootstrapped to achieve greater efficiency for simulating one-sided error randomized algorithms as well. The transformation used there differs in a subtle but crucial way from the one given here. Those results have been further extended to the problem of estimating the bias of an arbitrary population using a semi-random source for picking the samples [VV2]

The spread of the output distribution of a pseudo-random number generator provides a natural measure of its usefulness in simulating randomized algorithms. Analyzing the spread of generators such as the linear congruential generator should prove to be an interesting challenge.

5. Acknowledgements

We wish to thank Manuel Blum, Michael Fredman, Richard Karp and Rakesh for several stimulating discussions.

5. References

[AKS] M. Ajtai, Komlos, Szemeredi, "Deterministic Simulation of Log-space," 19th ACM Symposium on the Theory of Computing, 1987

[AR] N. Alon and M.O. Rabin, ''On the Random Properties of a Weakly Random Source,'' to appear.

[Bl] M. Blum, "Independent Unbiased Coin Flips From a Correlated Biased Source: a Finite State Markov Chain," 25th. IEEE Symposium on the Foundations of Computer Science, 1984, pp.425-433.

[BM] M. Blum and S. Micali, "How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits," SIAM J. on Computing, vol. 13 (1984), pp 270-299.

[CG] B. Chor and O. Goldreich, ''Unbiased Bits from Weak Sources of Randomness,'' 26th. IEEE Symposium on the Foundations of Computer Science, 1985. FOCS 1985.

[FKL] A.M. Frieze, R. Kannan, and J.C. Lagarius, "Linear Congruential Generators do not Produce Random Sequences," 25th. IEEE Symposium on the Foundations of Computer Science, 1984, pp. 480-484.

[Gi] J. Gill, ''Computational Complexity of Probabilistic Turing Machines,'' SIAM J. Comput. 6 (1977) pp. 675-695.

[Kn] D. Knuth, The Art of Computer Programming, Volume 2: Seminumerical Algorithms, Addison-Wesley, Reading, MA (second edition 1981).

[Pl] J. Plumstead, 'Inferring a Sequence Generated by a Linear Congruence," 23rd. IEEE Symposium on the Foundations of Computer Science, 1983, pp. 153-159.

[Sa] M. Santha, "On Using Deterministic Functions to Reduce Randomness in Probabilistic Algorithms," Information and Computation 74, 241-249 (1987).

[SV] M. Santha and U. V. Vazirani, "Generating Quasi-random Sequences from Slightly-random Sources," Journal of Computer and System Sciences, vol 33, No. 1, 75-87 (1986).

[Va1] U.V. Vazirani, Towards a Strong Communication Complexity Theory or Generating Quasi-Random Sequences from Two Communicating Slightly-random Sources,'' Combinatorica, vol 7, No. 4, 375-392 (1987).

[Va2] U.V. Vazirani, "Randomness, Adversaries and Computation," Ph.D. Dissertation (1986) U. C. Berkeley.

[Va3] U.V. Vazirani, "Efficiency considerations in using Semi-random Sources," 19th ACM Symposium on the Theory of Computing, 1987

[Va4] U.V. Vazirani, "Efficient Simulation of Randomized Algorithms with a Semi-random Source" to appear.

[VV1] U. V. Vazirani and V. V. Vazirani, "Random Polynomial Time is Equal to Semi-Random Polynomial Time," Symp on Foundations of Computer Science, 1985.

[VV2] U.V. Vazirani and V.V. Vazirani, "Sampling a Population with a Semi-random Source", Proceedings of FSTTCS Conference, (1986), pp 443-452.

[Ya] A. Yao, "Theory and Applications of Trapdoor Functions," 23th. IEEE Symposium on the Foundations of Computer Science, 1982.