How strong is Nisan's pseudo-random generator?

Matei David   Periklis A. Papakonstantinou   Anastasios Sidiropoulos

August 28, 2010

Abstract

We study the resilience of the classical pseudo-random generator (PRG) of Nisan [Nis92] against space-bounded machines that make multiple passes over the input. Our motivation comes from the derandomization of BPNC^1. Observe that if for every log-space machine that reads its input $n^{O(1)}$ times there is a PRG fooling this machine, then in particular we fool NC^1 circuits. Nisan's PRG is known to fool log-space machines that read the input once. We ask what are the limits of this PRG regarding log-space machines that make multiple passes over the input. We show that for every constant k Nisan's PRG fools log-space machines that make $\log^k n$ passes over the input, using a seed of length $\log^{k'} n$, for some $k' > k$. We complement this result by showing that in general Nisan's-PRG cannot fool log-space machines that make $n^{O(1)}$ passes even for a seed of length $2^{\Theta(\sqrt{\log n})}$. The observations made in this note outline a more general approach in understanding the difficulty of derandomizing BPNC^1.

1 Introduction and preliminaries

The seminal work of Nisan [Nis92] gives a PRG which fools log-space machines that read their input *once*. It is easy to see that $\mathsf{NC}^1 \subseteq \mathsf{L}$, e.g. [AB09]. However $\mathsf{BPNC}^1 \subseteq \mathsf{BPL}$ is not known to be true, where BPNC^1 and BPL are the standard definitions of the two-sided error randomized analogs of NC^1 and L. By definition BPL is characterized by log-space machines that read their randomness only once. On the other hand a family of probabilistic circuits roughly corresponds to a space bounded machine that reads its randomness *multiple times*.

To date there are explicit constructions of pseudo-random generators (PRGs) that fool space-bounded machines, which make a single pass over the input. A natural question is whether these PRGs already have the fooling power to show e.g. $\mathsf{BPNC}^1 \subseteq \mathsf{QuasiP}$.

Prior to our work Impagliazzo, Nisan and Wigderson [INW94] constructed a PRG, more complicated than the original PRG of Nisan (N-PRG), and they showed that it fools log-space machines for some number of passes over the input. We use a simple averaging argument to show that the original N-PRG already has this property[1] (Theorem 2.4).

Our second contribution is a uniform log-space and $n^{O(1)}$-passes distinguisher against N-PRG, which also answers our derandomization question in the negative (Theorem 3.2). It seems interesting to understand further whether this distinguisher can be generalized so as to put new derandomization questions in perspective - see Section 4 for a short discussion.

---

[1] To the best of our knowledge, this property of Nisan's PRG was not previously known (also: Noam Nisan, personal communication).

Preliminaries and notation. Derandomizing space and time-bounded classes is an area of intense interest; cf. the somewhat older but excellent surveys by Saks [Sak96] and Kabanets [Kab02].

We use standard names for complexity classes NC, BPNC, BPL; see e.g. [AB09]. In particular, we denote by $\mathsf{QuasiP} := \cup_{c>0} \mathsf{DTIME}(2^{\log^c n})$, BPNC^1 denotes the class of sets decidable by probabilistic uniform polynomial size circuits of bounded fan-in, and BPL denotes the class of sets decidable by probabilistic Turing Machines that read their random tape once.

$\mathsf{BP}^*\mathsf{L}$ denotes [Nis93] the class of sets decidable by log-space machines with an auxiliary read-only, polynomially long, and two-way tape. In particular, $\mathsf{BPNC}^1 \subseteq \mathsf{BP}^*\mathsf{L}$. We define $\mathsf{BPL}[r]$ to be the class of languages decidable by log-space machines with at most $r(n)$ passes over the randomness, and 2-sided error bounded away from $1/2$. Observe that a log-space Turing Machine with a polynomially-long, two-way random tape works in time $n^{O(1)}$, given that it always halts. Thus,

$$\mathsf{BPNC}^1 \subseteq \mathsf{BP}^*\mathsf{L} = \mathsf{BPL}[n^{O(1)}] := \cup_{k>0} \mathsf{BPL}[n^k]$$

We also know that $\mathsf{NC}^1 \subseteq \mathsf{L} = \mathsf{BPL}[0]$. In this sense, derandomizing along $\mathsf{BPL}[r]$ indicates progress towards the derandomization of BPNC^1. Also, let $\mathsf{BPL}[\mathsf{polylog}] := \cup_{k>0} \mathsf{BPL}[\log^k n]$.

Our results. In Section 2 we observe (Theorem 2.4) that a PRG resilient against multiple passes corresponds to a PRG with larger seed-length that fools single-pass machines. In Section 3 we complement this result by presenting a *uniform* log-space distinguisher which distinguishes N-PRG-strings even for *seed of length* $2^{c\sqrt{\log n}}$, for any $c > 0$. This holds for the standard family[2] of hash functions for N-PRG. Our distinguisher uses some elementary algebraic properties. It relies on a deep result due to Mulmuley [Mul87] (which derandomizes Borodin, Gathen and Hopcroft [BGH82]) for solving a non-singular system of linear equations with polynomial size, constant fan-in circuits of $O(\log^2 n)$ depth, which in particular implies a $O(\log^2 n)$ space algorithm. Our distinguisher works in logarithmic space - it uses the algorithm of [Mul87] in linear systems of size $O(2^{c\sqrt{\log n}})$, where n is the length of the whole input.

2 $\mathsf{BPL}[\log^{O(1)} n] \subseteq \mathsf{QuasiP}$: Nisan's PRG against poly-log passes

We show that using in a "black-box" way the analysis of N-PRG we can trade seed-length for passes over the input. Intuitively, this becomes possible since [Nis92] proves that it is not only hard to distinguish between the computation of a log-space machine on a random and on a pseudo-random tape, but more generally hard to distinguish between "partial computations" i.e. when we start at an arbitrary state and end to an arbitrary state.

Notation and preliminaries for N-PRG. We identify an one-way, non-uniform Turing Machine (i.e. TM with advice) M by a family of *Finite State Machines* (FSMs).

Let $G : A \to B$ be an arbitrary function, and let D be a probability distribution on A. We denote by $G(D)$ the probability distribution of the random variable $G(x)$, where x is chosen from D.

---

[2]Note that *some* restriction on the complexity of the family of hash functions is necessary for a distinguisher to exist.

Let $N \geq 1$, and let D be a probability distribution over $\{0,1\}^N$. For any $t \geq 1$, we denote by D^t the probability distribution over $\{0,1\}^{Nt}$ obtained by taking t concatenated copies of a string x that is chosen from D.

Let D_1, D_2 be probability distributions over a finite set A. We use the notation

$$\|D_1 - D_2\|_1 = \sum_{x \in A} \left| \Pr_{y_1 \in D_1} [y_1 = x] - \Pr_{y_2 \in D_2} [y_2 = x] \right|.$$

Let Q be a FSM with alphabet $\{0,1\}$, and with a designated initial state, and let D be a probability distribution over $\{0,1\}^n$. Then, we denote by $Q(D)$ the matrix whose (i,j)-th entry is the probability getting from state i to state j after reading a string x chosen from D.

For a matrix $P \in \mathbb{R}^{N \times N}$ let $\|P\|_1 = \max_i \sum_j |P_{i,j}|$.

Definition 2.1 (Pseudo-random generator against FSMs). Let $G : \{0,1\}^m \to \{0,1\}^n$ be computable in time polynomial in n, and let $\varepsilon, w > 0$. We say that G is a *pseudo-random generator against FSMs* for space w with parameter ε if for any FSM Q with 2^w states, we have $\|Q(U_n) - Q(G(U_m))\|_1 \leq \varepsilon$.

Theorem 2.2 (Nisan [Nis92]). *There exists $c > 0$ such that for any $N > 0$, there exists a function $G : \{0,1\}^{N^2} \to \{0,1\}^{N 2^{cN}}$ computable in time polynomial in 2^{cN} which is a pseudo-random generator against FSMs for space cN, with parameter 2^{-cN}.*

Our "seed-length-buys-passes" theorem. Instead of considering a distinguisher as an FSM with multiple passes over the input, it is convenient to consider a one-way FSM whose input contains multiple copies of the original input. For a function $G : \{0,1\}^m \to \{0,1\}^n$ we define $G^2 : \{0,1\}^m \to \{0,1\}^{2n}$, $x \mapsto G(x) \circ G(x)$, where $\circ$ denotes concatenation of strings. The following simple lemma states that N-PRG works (with a small loss in the parameter) even if the space-bounded machine is allowed to make two passes over the random tape.

Lemma 2.3. *Let $G : \{0,1\}^m \to \{0,1\}^n$ be a PRG against FSMs for space $2s$ with parameter ε. Then $G^2 : \{0,1\}^m \to \{0,1\}^{2n}$ is a PRG against FSMs with space s with parameter $\varepsilon' = \varepsilon \cdot 2^{2s}$.*

Proof. Suppose that G^2 is not pseudo-random against FSMs of space s with parameter ε'. We show that G is not a PRG against FSMs for space $2s$ with parameter ε.

Therefore, there exists a FSM with state space $\{1, \ldots, 2^s\}$ such that

$$\|Q(G^2) - Q(U_n^2)\|_1 > \varepsilon' \tag{2.1}$$

Let 1 be the initial state of Q. For a string $x \in \{0,1\}^n$, and for $i, j \in \{1, \ldots, 2^s\}$ we write $i \xrightarrow{x}_Q j$ to denote the event that starting from state i and reading the string x, the FSM Q ends up to state j. For any $i, j \in \{1, \ldots, 2^s\}$ let

$$p_{i,j} = \Pr_{x \in G}[1 \xrightarrow{x}_Q i \text{ and } i \xrightarrow{x}_Q j]$$

and

$$q_{i,j} = \Pr_{y \in U_n}[1 \xrightarrow{y}_Q i \text{ and } i \xrightarrow{y}_Q j]$$

3

From (2.1) we have

$$\sum_{i\in\{1,\ldots,2^s\}}\sum_{j\in\{1,\ldots,2^s\}}|p_{i,j}-q_{i,j}|\geq\|Q(G^2)-Q(U_n^2)\|_1>\varepsilon'$$

Therefore, there exist $i^*,j^*\in\{1,\ldots,2^s\}$ such that

$$|p_{i^*,j^*}-q_{i^*,j^*}|>\varepsilon'/2^{2s}$$

Let Q' be a FSM with state space $\{1,\ldots,2^s\}\times\{1,\ldots,2^s\}$, and alphabet $\{0,1\}$. We define the transition function of Q' so that for any $b\in\{0,1\}$ and for any $i,j\in\{1,\ldots,2^s\}$, we have $(i,j)\xrightarrow{b}_{Q'}(i',j')$ iff $i\xrightarrow{b}_Q i'$ and $j\xrightarrow{b}_Q j'$. We set the initial state of Q' to be $(1,i^*)$. We have

$$
\begin{aligned}
\|Q'(G)-Q'(U_n)\|_1 &= \sum_{i,j\in\{1,\ldots,2^s\}}\left|\Pr_{x\in G}[(1,i^*)\xrightarrow{x}_{Q'}(i,j)]-\Pr_{y\in U_n}[(1,i^*)\xrightarrow{y}_{Q'}(i,j)]\right|\\
&\geq \left|\Pr_{x\in G}[(1,i^*)\xrightarrow{x}_{Q'}(i^*,j^*)]-\Pr_{y\in U_n}[(1,i^*)\xrightarrow{y}_{Q'}(i^*,j^*)]\right|\\
&= \left|\Pr_{x\in G}[1\xrightarrow{x}_Q i^*\text{ and }i^*\xrightarrow{x}_Q j^*]-\Pr_{y\in U_n}[1\xrightarrow{y}_Q i^*\text{ and }i^*\xrightarrow{y}_Q j^*]\right|\\
&= |p_{i^*,j^*}-q_{i^*,j^*}|\\
&> \varepsilon'/2^{2s}\\
&= \varepsilon
\end{aligned}
$$

Therefore, G is not a PRG against FSMs of space $2s$ with parameter ε. $\qquad\square$

Note that every time we apply Lemma 2.3 on a PRG G we get a new PRG G' which is secure against twice as many passes as G, with slightly worse space and error parameters. By repeating $O(\log\log n)$ times, we obtain the following Theorem 2.4.

Theorem 2.4. *BPL*[polylog] $\subseteq$ *QuasiP*.

3 Breaking Nisan's PRG with polynomially many passes

Let n be the length of the output of N-PRG. Fix $c>0$ and the seed length to be $2^{c\sqrt{\log n}}$. In Theorem 2.2 we stated a corollary of the theorem from [Nis92], by choosing the parameters appropriately. We used Theorem 2.2 to obtain a positive result, which is independent of the structure of the PRG. However, for breaking N-PRG we consider its exact form.

Note, that our adversary overwhelmingly breaks the security properties (stated for one-pass) of N-PRG. In particular, for seed length $2^{c\sqrt{\log n}}$ instead of a $2^{c'\sqrt{\log n}}$-space distinguisher, $c'>0$, we present a *log-space* one. Furthermore, we distinguish with probability much bigger than $2^{-c'\sqrt{\log n}}$.

Let us review the description of N-PRG.

Let H be a universal family of hash functions $h:\{0,1\}^N\to\{0,1\}^N$, for some N. We are going to determine all parameters after the description of the PRG. For every integer $k\geq 0$ define the generator

$$G_k:\{0,1\}^N\times H^k\to(\{0,1\}^N)^{2^k}$$

where G_k is defined recursively as follows:

- $G_0(x) = x$, and

- $G_k(x, h_1, \ldots, h_k) = G_{k-1}(x, h_1, \ldots, h_{k-1}) \circ G_{k-1}(h_k(x), h_1, \ldots, h_{k-1})$, where $\circ$ denotes concatenation of strings.

Nisan shows that such a PRG is secure against FSMs that make a single pass over the input. He uses an explicit efficiently computable family H of affine functions $h : GF(2)^N \to GF(2)^N$. A function h is affine if there is a linear function f_h and a $b \in GF(2)^N$ such that $h(x) = f_h(x) + b$ for all $x \in GF(2)^N$. Nisan's [Nis92] family H has the property that each $h \in H$ can be described in $O(N)$ bits.

Additional terminology and notation. The output of the PRG consists of *blocks*, each of which is an N-bit binary string, and it corresponds to evaluating x on some composition of functions from H. Let $h_{i_1,\ldots,i_l} = h_{i_1} \circ \ldots \circ h_{i_l}$, where $\circ$ denotes function composition. By definition, in $G_k(x, h_1, \ldots, h_k)$ each block will be the composition of $h_{i_1} \circ h_{i_2} \circ \cdots \circ h_{i_l}$ evaluated on x, where $i_1 < i_2 < \cdots < l_l$. Also, given the output y of the PRG on some x with family $H = \{h_1, \ldots, h_k\}$, an increasing sequence $\langle i_1, i_2, \ldots i_l \rangle$, $i_j \in \{1, \ldots, k\}$ uniquely determines the position of $h_{i_1,\ldots,i_l}(x)$ in y.

The parameters. The output length is $n = 2^k N$. Therefore, $k = O(\log n)$. For a family H as in [Nis92], the seed length is $O(kN)$. Therefore, for seed length $2^{c\sqrt{\log n}}$ we have that $N = O(2^{c\sqrt{\log n}})$.

The main idea. Suppose that the string y is the output of N-PRG, with $H = \{h_1, \ldots, h_k\}$ the family of affine functions and some x. We will describe a test that every such y passes, but almost every string fails to pass. Note that to perform the test we do not need to know the actual h_i. The given $\langle i_1, \ldots, i_l \rangle$ is sufficient to determine the position of $h_{i_1,\ldots,i_l}(x)$ on y.

Here is a property that our test will use. We treat every block of y, which is in particular a binary substring of length N, as a vector in $GF(2)^N$. Let $y_1, \ldots, y_l$ be an even number of blocks in y and $h : GF(2)^N \to GF(2)^N$ an affine function. If $y_1, \ldots, y_l$ are linearly dependent then $h(y_1) + \cdots + h(y_l) = 0$. This holds since (i) f_h is a linear map, and thus in particular a homomorphism, and (ii) for an even l, $h(y_1) + \cdots + h(y_l) = f_h(y_1) + \cdots + f_h(y_l) = f_h(y_1 + \cdots + y_l)$. Furthermore, suppose that $h := h_1 \in H$ and h_1 is not used in the composed functions determining the y_i's. Recall that for every block y_i we know the position of $h(y_i)$ on y.

Therefore, the output of N-PRG y passes the following test:

> find an even number of linearly independent blocks $y_1, \ldots, y_l$ indexed by the h_i's where $i > 1$, and check whether the sum of the blocks corresponding to $h_1(y_i)$ evaluates to 0.

Observe that to perform this test we don't have to know the actual h_i's. We refer to the h_i's just for determining the positions of the corresponding blocks.

Observe that if y is random then the probability that at least 2 vectors corresponding to N-bit substrings of y sum up to 0 is 2^{-N}.

The log-space distinguisher. Let y be the input to the distinguisher.

1. Let $I_1 := \{2, 3, \ldots, \frac{k}{2}\}$, and $I_2 := \{\frac{k}{2} + 1, \ldots, k\}$. Use I_1 and I_2 to index block positions. Let H_1 be the first $N + 1$ blocks indexed by $\langle i_1, \ldots, i_l \rangle$, $1 < i_1 < \ldots < i_l$, with $i_j \in I_1$. Similarly,

we define H_2 where the indices are determined using I_2. Hence, in both cases for y (either y is the output of N-PRG on random H and x, or y is a random string) we have that (i) $H_1 \cap H_2 = \emptyset$ with probability $\geq 1 - 2^{-\Omega(N)}$, and (ii) each of the sets is larger than N, which guarantees that the vectors in H_1 are linearly dependent, and the same holds for H_2.

2. Use [Mul87] (see below) to find in space $O(\log n)$ a set of linearly dependent vectors in H_1: $y_1^{(1)} + y_2^{(1)} + \ldots + y_{j_1}^{(1)} = 0$, and a set of linearly dependent vectors in H_2: $y_1^{(2)} + y_2^{(2)} + \ldots + y_{j_2}^{(2)} = 0$. As usual in space-bounded computation, by this we mean that each time we can (re)compute the indices of the vectors on the input tape.

3. If j_1 is even, accept iff $\sum_{j=1}^{j_1} \alpha_{y_j^{(1)}} = 0$, where $\alpha_{y_j^{(1)}}$ is the block indexed by the composition of h_1 with the composed functions corresponding to $y_j^{(1)}$. Similarly, if j_2 is even. Else, if both j_1 and j_2 are odd then $j_1 + j_2$ is even, and accept iff $\sum_{j=1}^{j_1} \alpha_{y_j^{(1)}} + \sum_{j=1}^{j_2} \alpha_{y_j^{(2)}} = 0$.

Clearly, if the input is random then the probability that the distinguisher accepts is $2^{-\Omega(N)}$. On the other hand, if the input is pseudo-random then with probability at least $1 - 2^{-\Omega(N)}$ we have that (i) $H_1 \cap H_2 = \emptyset$, and (ii) all vectors in $H_1 \cup H_2$ are non-zero. Conditioned on these two events, the distinguisher accepts with probability 1.

Finding the linear dependencies. Let NONSINGULAR-EQUATIONS(n) be the problem of solving a non-singular $n \times n$ system of linear equations [BGH82].

Theorem 3.1 (Mulmuley [Mul87]). NONSINGULAR-EQUATIONS(n) *can be computed in the functional analog of the uniform* NC^2 *(i.e. the circuits can have multiple outputs).*

Consider the following procedure. Fix $y \in H_1$ and let A be the $N \times N$ matrix with columns $\{z \in H_1 - \{y\}\}$. If $y \neq 0$, then every solution of the system $Ax = y$ is non-zero. Since the set of vectors in H_1 is linearly dependent, there exists at least one y such that the system has a solution. This solution gives us the required set of vectors $y_1^{(1)} + \ldots + y_{j_1}^{(1)} = 0$. Similarly for H_2.

Recall that $\mathsf{NC}^2 \subseteq \mathsf{DSPACE}(\log^2 n)$. However, when solving the linear system $Ax = y$ in the above procedure, the input length is not n, but polynomial in $N = O(2^{c\sqrt{\log n}})$. Therefore, by Theorem 3.1, the above procedure can be implemented in space $O(\log n)$ and thus we obtain the following theorem. Note that the length of the advice is $O(2^{c\sqrt{\log n}})$.

Theorem 3.2. *Let $c > 0$. There exists $k > 0$, such that N-PRG with seed of length $2^{c(\sqrt{\log n})}$ is not secure against log-space machines that make n^k passes over the input.*

4 Discussion

Constructing a PRG that unconditionally fools two-way log-space machines is a difficult task. If such a reasonably efficient to compute (e.g. in polynomial time) PRG exists then we obtain $\mathsf{L} \subsetneq \mathsf{NP}$, a somewhat strong lower bound. However, there are more modest goals to pursue. It seems interesting to understand what is the main issue behind multiple passes. For example, is it possible to construct a PRG that fools log-space machines that make e.g. n^ε passes for some *fixed* $\varepsilon > 0$? Can we formulate an interesting family of Nisan-like PRGs and abstract out the property that makes them vulnerable to multiple passes? Furthermore, there are conceivably simpler related

questions. For example, can we construct a PRG G that fools a log-space machine when reading its input once, but G does not fool e.g. two-passes log-space machines?

Acknowledgements

References

[AB09] S. Arora and B. Barak. *Computational Complexity: A Modern Approach.* Cambridge University Press, 2009.

[BGH82] A . Borodin, J. von zur Gathen, and J. E. Hopcroft. Fast parallel matrix and GCD computations. *Information and Control*, 52(3):241–256, 1982. (also FOCS'82).

[INW94] R. Impagliazzo, N. Nisan, and A. Wigderson. Pseudorandomness for network algorithms. In *ACM Symposium on Theory of Computing (STOC)*. ACM Press, 1994.

[Kab02] V. Kabanets. Derandomization: a brief overview. *Bulletin of the EATCS*, 76:88–103, 2002.

[Mul87] K. Mulmuley. A fast parallel algorithm to compute the rank of a matrix over an arbitrary field. *Combinatorica*, 7(1):101–104, 1987. (also STOC'86).

[Nis92] N. Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992. (also STOC'90).

[Nis93] N. Nisan. On read-once vs. multiple access to randomness in logspace. *Theor. Comput. Sci.*, 107:135–144, 1993. (also Structure in Complexity Theory '90).

[Sak96] M. Saks. Randomization and derandomization in space-bounded computation. In *Proceedings of the 11th Annual IEEE Conference on Computational Complexity (CCC-96)*, pages 128–149, Los Alamitos, May 24–27 1996. IEEE Computer Society.