CS 6815 Pseudorandomness and Combinatorial Constructions **Fall 2018**

Lecture 18: October 30

Lecturer: Eshan Chattopadhyay *Scribe: Jesse Goodman*

In which we obtain pseudorandom generators for space-bounded computation, using constructions from Nisan and Nisan-Zuckerman.

18.1 Recap: space-bounded computation

In the last lecture, we introduced the following complexity classes:

- $\mathsf{DSPACE}(S(n))$: decision problems solvable by a Turing machine restricted to $O(S(n))$ space.
- $\mathsf{BPSPACE}(S(n))$: decision problems solvable by a probabilistic Turing machine (with two-sided bounded error) restricted to $O(S(n))$ space.

An important open problem is proving that $\mathsf{BPSPACE}(\log n) \subseteq \mathsf{DSPACE}(\log n)$; i.e., derandomizing logspace-bounded computation. Here, the current state-of-the-art is that $\mathsf{BPSPACE}(\log n) \subseteq \mathsf{DSPACE}(\log^{3/2} n)$, proven by Saks and Zhou [SZ99].

In order to derandomize a class $\mathsf{BPSPACE}(S(n))$, it is often convenient to model a probabilistic Turing machine as a family of objects called *branching programs.* An (R, w)*-read once branching program* (ROBP) is an acyclic directed graph with R *layers.* Each layer consists of w nodes, and each node (in the first $R-1$ layers) has an edge labeled $\mathsf{0}$ and an edge labeled $\mathsf{1}$ into nodes of the next layer. Each node in the last layer is labeled accept or reject. Finally, there is a special start node (outside of all layers) with two edges labeled $\mathsf{0}$ and $\mathsf{1}$ into the first layer.

An (R, w)-ROBP can be interpreted as a function $B : \{0,1\}^R \to \{0,1\}$ in the natural way: given an input $x \in \{0,1\}^R$, initiate a walk from the start node and use the bits of x in succession to decide which outgoing edge should be used to travel from the current node to a node in the following layer. After using all the bits in x, the walk will be on a node v in the final layer, and B will output 1 if v is labeled accept, and 0 otherwise.

A probabilistic Turing machine M used to solve a decision problem in $\mathsf{BPSPACE}(S(n))$ can be modeled by a family of ROBPs: for each $n \in \mathbb{N}$ and $x \in \{0,1\}^n$, we may construct an $(R \leq 2^{O(S(n))}, w \leq 2^{O(S(n))})$-ROBP $B_{M,x}$ that simulates the probabilistic behavior of M on input x. Each node in $B_{M,x}$ corresponds to a state of $M_x(r)$ (the machine M with input x hardcoded in, acting on random input r). The transitions (edges) in $B_{M,x}$ correspond to reading bits from the random string $r \in \{0,1\}^{2^{O(S(n))}}$. Note that we may bound the length of r as such because otherwise M must hit a state twice and potentially loop forever, contradicting the fact that M solves a problem in $\mathsf{BPSPACE}(S(n))$.

Thus, in order to show $\mathsf{BPSPACE}(S) \subseteq \mathsf{DSPACE}(T)$, it suffices to construct a T-space-computable pseudo-random generator (PRG) $G : \{0,1\}^T \to \{0,1\}^{2^S}$ that ϵ-fools $(2^S, 2^S)$-ROBPs, for small ϵ, e.g., 1/10. That is, for each such ROBP B, we must have $|\mathbb{E}[B(G(U_T))] - \mathbb{E}[B(U_{2^S})]| \leq \epsilon$. Given a machine M that decides a problem in $\mathsf{BPSPACE}(S)$, and an input x, we can then decide (deterministically) if x is a yes or no instance by simply taking the majority output from $M_x(G(z))$ over all $z \in \{0,1\}^T$.

In the following two sections, we will derandomize two space-bounded complexity classes using the above techniques with PRGs for ROBPs constructed by Nisan and Nisan-Zuckerman.

18.2 Nisan's PRG

In this section, we study the following classical pseudorandom generator of Nisan:

Theorem 18.1 ([Nis92]) *For any $R, w \in \mathbb{Z}^+, \epsilon > 0$, there exists an explicit PRG $G : \{0,1\}^{O((\log R)(\log(Rw/\epsilon)))} \to \{0,1\}^R$ that ϵ-fools (R, w)-ROBPs and that can be computed using space linear in the seed.*

As per the discussion above, we can use a family of (R, w)-ROBPs, with $R \leq n, w \leq n$, to simulate any machine deciding a problem in $\mathsf{BPSPACE}(\log n)$. Without loss of generality, we can assume $R = n, w = n$. Thus, observe that for a small fixed $\epsilon > 0$, Theorem 18.1 gives the generator $G : \{0,1\}^{O(\log^2 n)} \to \{0,1\}^{2^{\log n}}$ that ϵ-fools (n, n)-ROBPs, thereby showing $\mathsf{BPSPACE}(\log n) \subseteq \mathsf{DSPACE}(\log^2 n)$, using the discussion in the previous section.

In the following two subsections, we prove Theorem 18.1 by constructing and analyzing G.

18.2.1 The construction

Nisan's main realization was that since an ROBP has bounded width, if we examine the middle layer of the ROBP, then for most nodes in that layer, the probability of hitting that node v on a random walk (dictated by the input bits to the ROBP) is not too low. In other words, if such a random walk had length $R/2$, then the uniform distribution $U_{R/2}$ *conditioned on* arriving at node v should not have too much less entropy than the pure uniform distribution $U_{R/2}$. As such, without introducing too much error to the distribution computed by the ROBP on purely uniform bits, we should be able to recycle the random bits that directed us to node v in order to complete the random walk through the ROBP.

In order to recycle the random bits from the first half of the walk, Nisan uses a *seeded extractor*. Furthermore, in order to minimize the total number of truly random bits needed to fool the ROBP, Nisan applies the above idea recursively. We make all of this more precise, below.

Consider any $R, w \in \mathbb{Z}^+, \epsilon > 0$. Now, for any $l \in \mathbb{Z}^+, \epsilon' > 0$ and sufficiently large $d \in \mathbb{Z}^+$, let

$$\mathsf{Ext}_l : \{0,1\}^{ld} \times \{0,1\}^d \to \{0,1\}^{ld}$$

be a $(ld - \log w - \log(1/\epsilon'), \epsilon')$-seeded extractor, and let

$$G_l : \{0,1\}^{ld} \to \{0,1\}^{2^l}$$

be a pseudorandom generator defined recursively by:

$$G_l(z \circ y) := \begin{cases} y_1 \circ y_2, & l = 1 \\ G_{l-1}(z) \circ G_{l-1}(\mathsf{Ext}_{l-1}(z, y)), & l > 1, \end{cases}$$

where $z \in \{0,1\}^{d(l-1)}, y \in \{0,1\}^d$, and $\circ$ denotes concatenation. Note that by Theorem 6.22 in [Vad12], there exists an explicit Ext_l for $d = O(\log w + \log(1/\epsilon'))$.

In order to obtain PRG G referenced in Theorem 18.1 for parameters R, w, ϵ, we simply set $G = G_{l^*}$ with $l^* = \log R$, and we set $\epsilon' = \epsilon/4^{l^*}$. We analyze this construction (i.e. the error produced by G) in the following subsection.

18.2.2 The analysis

We now consider a branching program B of width w and bound the error of Nisan's PRG. For a node v in the branching program and an integer $r \in \mathbb{Z}^+$, it will be convenient to let $B_{v,r} : \{0,1\}^r \to [w]$ denote the sub-branching program that starts at node v and walks r steps. Furthermore, it will be convenient to let $\mathsf{Lay}_{v,r}$ denote the nodes of the layer in the branching program reached by starting at node v and walking r steps. The correctness of the construction immediately follows from the following claim, after plugging in $v = \mathsf{start}$ and the parameters from the previous subsection.

Claim 18.2 *Given a branching program B of width w, any node v in the branching program, and any integer $l \in \mathbb{Z}^+$, we have:*

$$|B_{v,2^l}(U_{2^l}) - B_{v,2^l}(G_l(U_{ld}))| \leq \epsilon_l$$

where $\epsilon_l = 4^l \cdot \epsilon'$ and $|\cdot|$ denotes statistical distance, or half the ℓ_1 norm.

Proof: The proof is by induction on l, and by studying the following four *hybrid* distributions:

- $D_1 := U_{2^{l-1}} \circ U'_{2^{l-1}}$
- $D_2 := U_{2^{l-1}} \circ G_{l-1}(U'_{d(l-1)})$
- $D_3 := G_{l-1}(U_{d(l-1)}) \circ G_{l-1}(U'_{d(l-1)})$
- $D_4 := G_{l-1}(U_{d(l-1)} \circ G_{l-1}(\mathsf{Ext}(U_{d(l-1)}, U'_d))$

Observe that $D_1 = U_{2^l}$, and $D_4 = G_l(U_{dl})$. To show the claim, it therefore suffices to show that for each D_i, D_{i+1}, where $i \in [3]$, $B_{v,2^l}$ cannot distinguish between D_i and D_{i+1}. We proceed in showing each such step:

- $|B_{v,2^l}(D_1) - B_{v,2^l}(D_2)| = |\sum_{u \in \mathsf{Lay}_{v,2^{l-1}}} p(u) \cdot B_{u,2^{l-1}}(U_{2^{l-1}}) - \sum_{u \in \mathsf{Lay}_{v,2^{l-1}}} p(u) \cdot B_{u,2^{l-1}}(G_{l-1}(U_{d(l-1)}))|$, where we use a convex combination trick in that $p(u)$ is defined as the probability distribution over the vertices in $\mathsf{Lay}_{v,2^{l-1}}$ induced by $B_{v,2^{l-1}}(U_{2^{l-1}})$. We may continue by pulling the sum and $p(u)$ out of the norm to obtain $\sum_{u \in \mathsf{Lay}_{v,2^{l-1}}} p(u)|B_{u,2^{l-1}}(U_{2^{l-1}}) - B_{u,2^{l-1}}(G_{l-1}(U_{d(l-1)}))| \leq \epsilon_{l-1}$, by the induction hypothesis and definition of a probability distribution.
- $|B_{v,2^l}(D_2) - B_{v,2^l}(D_3)| \leq \epsilon_{l-1}$, using the exact same idea as above.
- We use a slightly modified version of the above idea for bounding the error between the last two distributions. We first define a set of nodes in $\mathsf{Lay}_{v,2^{l-1}}$ for which hitting such a node indicates that much entropy was lost. In particular, we set $\mathsf{Bad} := \{u \in \mathsf{Lay}_{v,2^{l-1}} : p(u) \leq \epsilon'/w\}$, where $p(u)$ is the probability distribution on layer $\mathsf{Lay}_{v,2^{l-1}}$ induced by $G_{l-1}(U_{d(l-1)})$. We observe that for any $u \notin \mathsf{Bad}$, $H_\infty(U_{d(l-1)} \mid B_{v,2^{l-1}}(G_{l-1}(U_{d(l-1)})) = u) \geq d(l-1) - \log w - \log(1/\epsilon')$. As such, our extractor will succeed whenever a walk ends on such a node. In conclusion, we have: $|B_{v,2^l}(D_3) - B_{v,2^l}(D_4)| = \sum_{u \in \mathsf{Lay}_{v,2^{l-1}}} p(u)|B_{u,2^{l-1}}(G_{l-1}(U'_{d(l-1)})) - B_{u,2^{l-1}}(G_{l-1}(\mathsf{Ext}(U_{d(l-1)}, U'_d)))| \leq \epsilon' + \sum_{u \in \mathsf{Bad}} p(u) \cdot 1 \leq \epsilon' + w \cdot (\epsilon'/w) \leq 2\epsilon'$.

Thus, we see that through the entire hybridization, our total error is $\epsilon_l \leq 2\epsilon_{l-1} + 2\epsilon'$, a recurrence which easily yields $\epsilon_l \leq 4^l \cdot \epsilon'$. ■

18.3 The Nisan-Zuckerman PRG

In this section, we study the Nisan-Zuckerman PRG:

Theorem 18.3 ([NZ96]) *For any $w \in \mathbb{Z}^+, R = \mathsf{polylog}(w)$, and small $0 < \gamma < 1$, there exists an explicit linear-space computable PRG $G : \{0,1\}^{O(R/(\log w)^\gamma)} \to \{0,1\}^R$ that ϵ-fools (R,w)-ROBPs for $\epsilon = 2^{-\Omega((\log w)^{1-\gamma}/\log^2 \log w)}$.*

Given the construction we are about to present, it is not difficult to show that by composing a constant number of these generators, we may produce a PRG $G^* : \{0,1\}^{O(\log w)} \to \{0,1\}^R$ that fools (R,w)-ROBPs with small ϵ, as long as $R = \mathsf{polylog}(w)$. Notice that unlike Nisan's generator, the final Nisan-Zuckerman generator G^* works in space $O(\log w)$ (instead of $O(\log^2 w)$), but only produces $\mathsf{polylog}(w)$ bits instead of up to w bits. Thus, we see that if we define $\mathsf{BPSPACE}(\log n, \mathsf{polylog}(n))$ as the problems in $\mathsf{BPSPACE}(\log n)$ that can be solved using at most $\mathsf{polylog}(n)$ random bits (i.e., polynomial in the space of the machine), then the Nisan-Zuckerman PRG derandomizes this complexity class! That is, it shows that $\mathsf{BPSPACE}(\log n, \mathsf{polylog}(n)) \subseteq \mathsf{DPSPACE}(\log n)$.

In the following two subsections, we prove Theorem 18.3 by constructing and analyzing G.

18.3.1 The construction

The Nisan-Zuckerman PRG uses the same idea of "bit-recycling" as Nisan's generator, but it does not generate a recursive tree. The construction is as follows: consider any $w \in \mathbb{Z}^+, R = \mathsf{polylog}(w)$, and small $0 < \gamma < 1$. Now, for any $n, m \in \mathbb{Z}+, \epsilon' > 0$ and sufficiently large $d \in \mathbb{Z}^+$, let

$$\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$$

be a $(n/2, \epsilon')$-seeded extractor, and for any $t \in \mathbb{Z}^+$, let

$$G_t : \{0,1\}^{n+td} \to \{0,1\}^{tm}$$

be a pseudorandom generator defined by:

$$G_t(x \circ y_1 \circ \cdots \circ y_t) = \mathsf{Ext}(x, y_1) \circ \cdots \circ \mathsf{Ext}(x, y_t),$$

where $x \in \{0,1\}^n$, each $y_i \in \{0,1\}^d$, and $\circ$ denotes concatenation. Note that by Theorem 6.36 in [Vad12], there exists an explicit Ext as described above with $d = O(\log(n/\epsilon'))$.

In order to obtain the PRG G referenced in Theorem 18.3 for parameters R, w, γ, we simply set $n = 4 \log w, d = (\log w)^{1-\gamma}, m = \log w$, and $t = R/\log w$.

We analyze this construction (i.e. the error produced by G) in the following subsection.

18.3.2 The analysis

Given a PRG G_t as defined in the previous subsection, let ϵ_t be its error and let ϵ' be the error of the extractors baked in. The correctness of the Nisan-Zuckerman construction immediately follows from the following claim, after plugging in $v = \mathsf{start}$ and the parameters from the previous subsection.

Claim 18.4 *Given a branching program B of width w, any node v in the branching program, and any integer $t \in \mathbb{Z}^+$, we have:*

$$|B_{v,tm}(U_{tm}) - B_{v,tm}(G_t(U_{n+td}))| \le \epsilon_t,$$

where $\epsilon_t = t(\epsilon' + 1/w)$.

Proof: The proof is by induction on t. We let q denote the distribution $B_{v,(t-1)m}(G_{t-1}(U_{n+(t-1)d}))$, and let p denote the probability distribution induced by $B_{v,(t-1)m}(U_{(t-1)m})$. By the induction hypothesis, $|p-q| \leq (t-1)(\epsilon' + 1/w)$. As with Nisan's generator, we will define a set $\mathsf{Bad} := \{u \in \mathsf{Lay}_{v,(t-1)m} : p(u) \leq 1/2^{2m}\}$. Observe that for $u \notin \mathsf{Bad}$, $H_\infty(U_{n+(t-1)d} \mid B_{v,(t-1)m}(G_{t-1}(U_{n+(t-1)d}))) \geq n - 2m$. It is then straightforward to decompose the distributions in the claim, apply the triangle inequality, and extract from the "good" nodes in order to obtain the desired result. ■

References

[Nis92] N. Nisan, Pseudorandom generators for space-bounded computation. *Combinatorica* **12.4** (1992): 449-461.

[NZ96] N. Nisan and D. Zuckerman, Randomness is linear in space. *Journal of Computer and System Sciences* **52.1** (1996): 255-262.

[SZ99] M. Saks and S. Zhou, $\mathsf{BP_H SPACE}(S) \subseteq \mathsf{DSPACE}(S^{3/2})$. *Journal of Computer and System Sciences* **58.2** (1999): 376-403.

[Vad12] S. Vadhan, Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science* **7.1-3** (2012): 1-336.