CS395T Pseudorandomness and Combinatorial Constructions
Lecture 16, March 8, 2001
Lecturer: Prof. David Zuckerman, Scribe: Vladimir Trifonov

Randomized Space-Bounded Computation

1 Turing Machine Model

Under the Turing Machine Model a deterministic computation is modeled by an abstract computational device (Turing Machine, TM) which has an input tape, finite control and constant number (≥ 1) of work tapes. We will assume that the tapes are infinite in only one direction. Each tape contains a sequence of symbols from a given finite alphabet Σ. Each tape has a head which can scan only one symbol at a time and can move in both directions. The head of the input tape is read-only and the heads of the work tapes are read-write. The finite control can be in one of a finite number of states. Based on the tapes' content scanned by the heads and its internal state the finite control can change its state, change the tape content under the heads and move the heads to adjacent cells.

A configuration of a TM is described by the contents of its tapes, the locations of the heads and its internal state. In the initial configuration every head is at the beginning of its tape, the input tape contains an encoding of the input on which the machine will perform a computation, and the work tapes are empty. Notice that if a computation of a TM halts in finite time then on each work tape only a finite number of cells are used. The space used by a computation of a TM M on input x is

$$S_M(x) = c \cdot \max\{S_i(x) : i \in [c]\},$$

where c is the number of tapes of M and $S_i(x)$ is the number of cells used on tape i by the computation of M on x. For any n the space of M on inputs of size n is $S_M(n) = \max\{S_M(x) : |x| = n\}$.

Since it is known that we can change the constant factors of S_M at the expense of increasing the number of states and the size of the alphabet of M, it is legal to use asymptotic notation when expressing S_M.

Let M be a TM with c work tapes and working in space $S(n)$. Then the total number of possible configurations C of M is $O(nS^c|\Sigma|^{cS}) = n2^{O(S)}$, since we have n possible positions of the input tape's head, S^c possible positions of the work tapes' heads and $|\Sigma|^{cS}$ possible contents of the work tapes. Hence $\log C = \log n + O(S)$ and as long as $S(n) \geq \log n$ we can say that $S = O(\log C)$.

To allow the usage of random bits by a TM we can assume that the finite control has access to a source of random bits – this could be either in the form of coin flips by the finite control or an additional 1-way read-only tape of random bits. The transition of the finite control now is determined not only by its current configuration, but by a random bit as well. Notice that the random bits are used in a read-once fashion – if we want to access a previous random bit we have to explicitly store it on a work tape. This allows us to consider separately the space and random bits requirements of the TM. For a randomized TM we will also have a requirement on the number of correct/incorrect computations on a given input. If the requirement is that we want to have at least one correct computation than we have a non-deterministic TM.

A deterministic TM using space $S \geq \log n$ has $2^{O(S)}$ configurations; hence, if the machine halts, it does so in time $2^{O(S)}$. For a randomized TM this is not necessarily true as shown by the following example. Consider the randomized TM which on input of size n flips random bits until it flips n consecutive 1's. The expected running time of this computation is 2^n, but the space needed by it is only $\log n$, because we only need to store the length of the current sequence of 1's. The running time of this machine could be double exponential in the space it needs. In fact if a bound on the running time of a randomized TM is not imposed, randomization becomes as powerful as non-determinism.

Definition 1 *Define the following complexity classes:*

- $\mathbf{DSPACE}(S) = \{L : L$ *is accepted by a deterministic TM in space* $S\}$,
- $\mathbf{NSPACE}(S) = \{L : L$ *is accepted by a non-deterministic TM in space* $S\}$,
- $\mathbf{RSPACE}(S) = \{L : L$ *is accepted by a randomized TM in space* S *and time* $\leq 2^S$, *with one-sided error*$\}$,
- $\mathbf{BPSPACE}(S) = \{L : L$ *is accepted by a randomized TM in space* S *and time* $\leq 2^S$, *with two-sided error*$\}$,
- $\mathbf{L} = \mathbf{DSPACE}(\log n)$, $\mathbf{NL} = \mathbf{NSPACE}(\log n)$, $\mathbf{RL} = \mathbf{RSPACE}(\log n)$, $\mathbf{BPL} = \mathbf{BPSPACE}(\log n)$ *and* $\mathbf{L}^k = \mathbf{DSPACE}(\log^k n)$, *for* $k \geq 1$.

Having an input x and a non-deterministic TM machine M running in space S on x we construct the configuration graph G_M of M in the following way. The vertices of G_M are the configurations of M and there is an edge from configuration C_1 to configuration C_2, if C_2 follows from C_1. It is immediate that G_M has 2^S nodes and M accepts x iff there is a path in G_M from the initial configuration of M to an accepting configuration.

Using the configuration graph of a space-bounded TM we have the following inclusion due to Savitch.

Theorem 1 $\mathbf{NSPACE}(S) \subseteq \mathbf{DSPACE}(S^2)$.

Corollary 1 $\mathbf{RSPACE}(S) \subseteq \mathbf{DSPACE}(S^2)$.

We also have the following inclusion due to Borodin, Cook and Pippenger and later Nisan.

Theorem 2 $\mathbf{BPSPACE}(S) \subseteq \mathbf{DSPACE}(S^2)$

The big open questions are:

- $\mathbf{RSPACE}(S) = \mathbf{DSPACE}(S)$?
- $\mathbf{BPSPACE}(S) = \mathbf{DSPACE}(S)$?

These questions are actually equivalent to the same questions for $S = \log n$. We can see this using a padding argument which works in the following way. For $L \subseteq \Sigma^*$ define $L' = \{x\Box^{2^{S(|x|)-|x|}} : x \in L\}$, where $\Box \notin \Sigma$ denotes a special empty symbol. Then assuming that $\mathbf{RL} = \mathbf{L}$, $L \in \mathbf{RSPACE}(S)$ iff $L' \in \mathbf{RL}$ iff $L' \in \mathbf{L}$ iff $L \in \mathbf{DSPACE}(S)$.

Currently the best result on derandomizing space-bounded computation is due to Saks and Zhou.

Theorem 3 $\mathbf{BPL} \subseteq \mathbf{L}^{3/2}$

2 s-t Connectivity

Problem
Input: Graph $G = (V, E)$, $n = |V|$, $m = |E|$, and two vertices $s, t \in V$.
Output: Is there a path in G from s to t?

When G is directed we have Directed s-t Connectivity, otherwise Undirected s-t Connectivity. Defining the configuration graph of a space-bounded computation we also informally proved the following result.

Theorem 4 *Directed s-t Connectivity is* **NL**-*complete.*

We will show the following result due to Aleliunas, Karp, Lipton, Lovasz and Rackoff.

Theorem 5 *Undirected s-t Connectivity is in* **RL**.

Let G be the undirected graph in question and $s, t \in V$. The randomized algorithm for Undirected s-t Connectivity is straightforward – do a random walk in G starting at s and in each step check if it has reached t. We will prove that if s and t are connected than after a polynomial number of steps the probability that we have reached t is at least $1/2$.

Let π be the stationary distribution of the random walk on G, i.e. $\pi_u = \deg(u)/(2m)$, for $u \in V$. For $u \in V$ define the following random variables

- H_u = the time at which the random walk first visits u.
- H_u^+ = the time at which the random walk first visits u excluding time 0.

For a random variable using a random walk on G denote with $E_u X$ the expected value of X if the random walk starts at $u \in V$. Analogously for $e \in \vec{E}$, where $\vec{E}$ is the set of directed edges corresponding to E, define H_e, H_e^+ and $E_e X$.

Proposition 1 *For every $u \in V$, $E_u H_u^+ = 1/\pi_u$.*

Idea of Proof: u appears π_u fraction of times, i.e. the probability that the random walk visits u at any time is π_u. Hence the expected time between two appearances is $1/\pi_u$. $\square$

Proposition 2 *For every $e = (u, v) \in \vec{E}$, $E_e H_e^+ = 2m = E_v H_e$.*

Idea of Proof: In the stationary distribution over V, a probability mass of $1/2m$ is exchanged along both directions of every edge. This mass is the stationary probability of every edge in $\vec{E}$. Now the argument is the same as in Proposition 1. $\square$

From Proposition 2 it immediately follows that for $u, v \in V$, if $\{u, v\} \in E$, then $E_u H_v + E_v H_u \leq 2m$ and in particular $E_u H_v < 2m$. In general if $\text{dist}(u, v)$ is the distance between u and v, $E_u H_v < 2m \cdot \text{dist}(u, v) < 2mn$. Theorem 5 follows by taking $u = s$ and $v = t$, and applying Markov's Inequality on random walk of $4mn$ steps.

3 Cover Time of Random Walk

Definition 2 *Define the following quantities for a random walk:*

- $h_{max} = \max_{u,v \in V} E_u H_v$,
- Cover time $cov = \max_{u \in V} E_u C$, *where C is the first time the random walk visits all vertices of G.*

Notice that we can compute h_{max} in polynomial time since we have a system of linear equations $E_u H_v = 1 + \frac{1}{\deg(u)} \sum_{w \in \Gamma(u)} E_w H_v$.

The following table shows the values of these quantities for random walks on some types of graphs.

Graph Type	Mixing Time	h_{max}	cov
Cycle	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$
2D Torus	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n \log^2 n)$
Hypercube	$\Theta(\log n \log \log n)$	$\Theta(n)$	$\Theta(n \log n)$
Expander	$\Theta(\log n)$	$\Theta(n)$	$\Theta(n \log n)$
Complete	$\Theta(1)$	n	$n(1 + \frac{1}{2} + \cdots + \frac{1}{n-1})$

Theorem 6 $cov \leq 2mn$.

Proof: Let $u \in V$. Find a spanning tree T of G rooted at u. The expected time to visit all vertices starting at u is at most the expected time in which the random walk starting at u visits all vertices of T in depth first order. Hence

$$cov \leq \sum_{\{v,w\} \in E(T)} (E_v H_w + E_w H_v) \leq (n-1)2m$$

The first inequality holds since the depth first traversal of T visits every edge exactly twice. The second inequality comes from Proposition 2 and the fact that T has $n-1$ edges. $\square$

Theorem 7

$$cov \leq h_{max}\left(1 + \frac{1}{2} + \cdots + \frac{1}{n}\right).$$

Proof: We will prove the weaker $cov = O(h_{max} \ln n)$. To see this, break the random walk into l intervals of length $2h_{max}$. For any $u \in V$, the probability that the random walk misses u in a particular interval is at most $1/2$. Hence the probability that the random walk misses u in all intervals is at most 2^{-l}. Therefore the probability that the random walk misses some vertex in all intervals is at most $n2^{-l}$. Taking $l = \log n + 1$ we obtain that the last probability is at most $1/2$. Thus $cov \leq 2 \cdot l \cdot 2h_{max} = O(h_{max} \ln n)$. $\square$

For a complete graph we can compute cov exactly in the following way. Assume that we count the starting vertex as visited. Then

$$
\begin{aligned}
cov &= E[\text{time to reach the 1st new vertex}] \\
&\quad +E[\text{time to reach the 2nd new vertex}] \\
&\quad \vdots \\
&\quad +E[\text{time to reach the } (n-1)\text{-st new vertex}] \\
&= \frac{n}{n-1} + \frac{n}{n-2} + \ldots + n \\
&= n\left(1 + \frac{1}{2} + \cdots + \frac{1}{n-1}\right)
\end{aligned}
$$