

CMPT 710/407 - Complexity Theory: Lecture 11

Valentine Kabanets

October 16, 2007

1 Savitch's Theorem

We'll prove an amazing result: Nondeterministic space algorithms can be simulated efficiently by deterministic space algorithms, with only quadratic loss in space usage. That is, nondeterminism does *not* give us extra power in the case of space-bounded computation!

Recall that $\text{NL} = \text{NSpace}(\log n)$, and $\text{NPSPACE} = \cup_k \text{NSpace}(n^k)$.

First, using the notion of configuration graphs, we can show the following.

Theorem 1. $\text{NL} \subseteq \text{P}$

Proof. The proof is exactly the same as that for $\text{L} \subseteq \text{P}$. □

Configuration graphs of space-bounded TMs are a very useful tool for analyzing space-bounded computation. As the next theorem shows a reachability problem for graphs exactly captures the complexity of the class NL .

Define $ST - CON = \{(G, s, t) \mid G \text{ is a directed graph with a path from } s \text{ to } t\}$.

Theorem 2. $ST - CON$ is NL -complete under logspace reductions.

Remark: Note that we restricted the class of reductions to logspace computable ones. This is necessary to make the notion NL -completeness nontrivial. (As you recall from Problem Set 1, basically every language in P is P -complete under *polytime* reductions.)

Proof. 1. $ST - CON \in \text{NL}$: Given a graph $G = (V, E)$ and $s, t \in V$, nondeterministically guess a path from s to t , by keeping track of a current vertex on a path and a next vertex on a path. After guessing a next vertex on a path, make it the new current vertex (erasing the old current vertex) so as to re-use the space, and run in $O(\log n)$ space only. If ever t is a current vertex on a guessed path, then Accept.

2. $ST - CON$ is NL -hard: Given any language $L \in \text{NL}$ decided by some logspace NTM M , and an input x , construct the configuration graph of M on x . This can be done in logspace (can you see why?) Make s to be the start configuration, and t the accepting configuration. (Note: we can always modify any given NTM so that it has only one accepting configuration: after entering q_{accept} , the machine will erase all of its work-tapes, and go to the first non-blank symbol of its input tape.) □

Theorem 3 (Savitch's Theorem). $ST - CON \in \text{Space}(\log^2 n)$.

Corollary 4. 1. $NL \subseteq \text{Space}(\log^2 n)$.

2. $\text{NPSpace} = \text{PSPACE}$.

Proof of Savitch's Theorem. We will design a $\log^2 n$ -space algorithm that, given a directed graph $G = (V, E)$ with $|V| = n$ nodes, and $s, t \in V$, will accept iff t is reachable from s . (For convenience, we assume that $(u, u) \in E$ for every node $u \in V$.)

We design algorithm $Path(x, y, i)$ which accepts iff there is a path from x to y of length at most 2^i . Note that t is reachable from s iff $Path(s, t, \log n)$ accepts. (Do you see why?)

Algo $Path(x, y, i)$

if $i = 0$ **then**

 Accept iff $(x, y) \in E$

end if

for every $z \in V$

if $Path(x, z, i - 1)$ accepts **AND**

$Path(z, y, i - 1)$ accepts % we re-use space here!

then Accept

end if

end for

 Reject % if no "middle" point z was found, we reject

end Algo

It is not hard to see that algorithm $Path$ is correct. To analyze the space used, note that the depth of the recursion is $\log n$, and that the size of each "stack record" during the recursion is the size of $(x, y, i) \in O(\log n)$. Thus, the total space used is $O(\log^2 n)$. \square

It is still a big open problem to decide if $NL = L$. To show this, it would suffice to give a deterministic logspace algorithm for ST-CON, the problem of st-connectivity for *directed* graphs on n vertices. Interestingly, Reingold [Rei05] has recently showed that the st-connectivity problem for *undirected* graphs is solvable in deterministic logspace! The algorithm for doing this is highly nontrivial and not at all obvious; it is based on the algebraic characterization of connectivity in graphs (in terms of the so-called eigenvalue gap, the difference between the two largest eigenvalues of the adjacency matrix of a given graph). This algorithmic success renewed the interest in the NL vs. L question.

Next we will see another surprising result showing that $NL = \text{coNL}$, something we don't expect to be true in the setting of time complexity classes. (This might be taken as another piece of evidence pointing to the possibility of $NL = L$...)

References

[Rei05] O. Reingold. Undirected st-connectivity in log-space. In *Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing*, pages 376–385, 2005.