CSE 200 Computability and Complexity Wednesday, April 24, 2013

Lecture 8: PSPACE-Completeness & Savitch's Theorem

Instructor: Professor Shachar Lovett *Scribe: Dongcai Shen*

1 Recap: Space Complexity

Recall the following definitions on space complexity we learned in the last class:

$\mathsf{SPACE}(S(n)) \stackrel{\text{def}}{=}$ "languages computable by a TM where input tape is readonly and work tape size $S(n)$"

$\mathsf{LOGSPACE} \stackrel{\text{def}}{=} \mathsf{SPACE}(O(\log n))$

$\mathsf{PSPACE} \stackrel{\text{def}}{=} \cup_{c \geq 1} \mathsf{SPACE}(n^c)$

For example, nowadays, people are interested in streaming algorithms, whose space requirements are low. So researching on space complexity can have real-world influence, especially in instructing people the way an algorithm should be designed and directing people away from some impossible attempts.

Definition 1 (Nondeterministic Space NSPACE) The *nondeterministic space* $\mathsf{NSPACE}(S(n)) \stackrel{\text{def}}{=}$

- Input tape, read-only.
- Proof tape, read-only & read-once.
- Work tape size $S(n)$.

So, we can define

- $\mathsf{NL} \stackrel{\text{def}}{=} \mathsf{NSPACE}(O(\log n))$.
- $\mathsf{NPSPACE} \stackrel{\text{def}}{=} \cup_{c \geq 1} \mathsf{NSPACE}(n^c)$. □

Theorem 2 $\texttt{PATH} \stackrel{\text{def}}{=} \{(G, s, t) : G$ *directed graph,* s, t *vertices, there is a path* $s \rightsquigarrow t$ *in* $G\}$ *is* NL*-complete under logspace reduction.*

Corollary 3 *If we could find a deterministic algorithm for* PATH *in* $\mathsf{SPACE}(O(\log n))$*, then* $\mathsf{NL} = \mathsf{L}$.

Remark. Nondeterminism is pretty useless w.r.t. space. We don't know whether $\mathsf{NL} = \mathsf{L}$ yet.

2 Outline of Today

Definition 4 (Quantified Boolean formula) A *quantified Boolean formula* is $\forall x_1 \exists x_2 \forall x_3 \exists x_4 \cdots \phi(x_1, \cdots, x_n)$ where ϕ is a CNF. Let $\texttt{TQBF} \stackrel{\text{def}}{=} \{$A true QBF$\}$.

Theorem 5 TQBF *is* PSPACE*-complete under poly-time reduction (and actually under logspace reductions).*

Theorem 6 (Savitch's Theorem [4]) $\mathsf{NSPACE}(S(n)) \subseteq \mathsf{SPACE}(S^2(n))$. □

Savitch's Theorem has two important direct implications concerning the relationship between deterministic space and nondeterministic space. Surprisingly, nondeterministic poly-space is the same as deterministic poly-space, while nondeterministic logspace can be solved in deterministic logspace square.[1].

[1]Notice L is not defined as $\cup_{c \geq 1} \mathsf{SPACE}(\log^c n)$ and NL is not defined as $\cup_{c \geq 1} \mathsf{NSPACE}(\log^c n)$. The log polynomial version is called *Steve's Class* in honor of Steve Cook: $\mathsf{polyL} \stackrel{\text{def}}{=} \mathsf{SC} \stackrel{\text{def}}{=} \cup_{c \geq 1} \mathsf{SPACE}(\log^c n)$ (See the text [1] Page 94 Exercise 4.12).

Corollary 7

- $\mathsf{NL} \subseteq \mathsf{SPACE}(O(\log(n)^2))$.
- $\mathsf{NPSPACE} = \mathsf{PSPACE}$. □

On the other hand, people know the following related results:

- UNDIRECTED-PATH (PATH in undirected graphs) is in L (Reingold [2]).
- Randomized logspace $\subseteq \mathsf{L}^{3/2}$ (Saks and Zhou [3]).

Definition 8 (co-NSPACE($\cdot$)) co-$\mathsf{NSPACE}(S(n)) \stackrel{\text{def}}{=} \{L : L^c \in \mathsf{NSPACE}(S(n))\}$. Complete problem for co-NL is $\{(G, s, t) : G$ directed graph, s, t vertices, there is no path $s \rightsquigarrow t\}$. □

3 TQBF is PSPACE-Complete

Theorem 9 ([5]) TQBF *is* PSPACE*-complete under poly-time reduction.*

Proof

Step #1: TQBF $\in$ PSPACE. Let $\psi = \overbrace{\exists}^{=Q_1} x_1 \overbrace{\forall}^{=Q_2} x_2 \overbrace{\exists}^{=Q_3} x_3 \forall x_4 \cdots \phi(x_1, \cdots, x_n)$. Need to compute ψ in poly-space. Let $\psi \stackrel{\text{def}}{=} Q_1 x_1 Q_2 x_2 \cdots \phi(x_1, \cdots, x_n)$ where $Q_i \in \{\exists, \forall\}$.

For $1 \le i \le n$ and values $a_1, \cdots, a_i \in \{0, 1\}$ define $f_i(a_1, \cdots, a_i) = Q_{i+1} x_{i+1} \cdots Q_n x_n \phi(a_1, \cdots, a_i, x_{i+1}, \cdots, x_n)$. We have that

- If $Q_{i+1} = \exists$, then $f_i(a_1, \cdots, a_i) = f_{i+1}(a_1, \cdots, a_i, 0) \vee f_{i+1}(a_1, \cdots, a_i, 1)$.
- If $Q_{i+1} = \forall$, then $f_i(a_1, \cdots, a_i) = f_{i+1}(a_1, \cdots, a_i, 0) \wedge f_{i+1}(a_1, \cdots, a_i, 1)$.

We want to compute $\phi = f_0$ and we know how to compute $f_n(a_1, \cdots, a_n) = \phi(a_1, \cdots, a_n)$. The intuition is that we will compute f_0 by exploring the tree of possible partial assignments by a DFS-like process. Formally, in order to compute $f_i(a_1, \ldots, a_i)$ we need to run two instances of f_{i+1}. Note that we can run them using the same space, and we will need $O(\log n)$ overhead to remember indices, etc. So, if we compute f_{i+1} is space S_{i+1} then we can compute f_i in space $S_i = S_{i+1} + O(\log n)$. Overall, we can compute ψ in space $O(n \log n)$, which is polynomial space.

Step #2: TQBF is PSPACE-hard. Fix any language $L \in \mathsf{PSPACE}$. Consider its configuration graph G. The number of nodes in G is $N = \exp(n^c)$ for some $c > 0$. We need to know if there is a path $v_{\text{start}} \rightsquigarrow v_{\text{accept}}$ between the start and accept configurations of L. We note that to check if an edge $(u, v) \in G$ can be expressed by CNF $\phi(u, v)$ of size poly(n).

A naïve try. To check if there is a path $v_{\text{start}} \rightsquigarrow v_{\text{accept}}$ we write an exponential-sized formula:

$$\exists v_1 \exists v_2, \cdots \exists v_N \ \phi(v_{\text{start}}, v_1) \wedge \phi(v_1, v_2) \wedge \cdots \wedge \phi(u_{i-1}, u_N) \wedge \phi(u_N, v_{\text{accept}}).$$

This is problematic since the formula is of exponential size, and we would like a polynomial size formula.

Attempt to reduce size. Let $\mathsf{reach}_i(u, v) \stackrel{\text{def}}{=}$"there is a path $u \rightsquigarrow v$ of length $\le 2^i$". Then

$$\mathsf{reach}_{i+1}(u, v) = \exists w \ \mathsf{reach}_i(u, w) \wedge \mathsf{reach}_i(w, v).$$

However, it's still of exponential size. The trick to get polynomial size is to use the $\forall$ quantifier.

Polynomial size: We can write $reach_{i+1}(u, v)$ as

$$\mathsf{reach}_{i+1}(u, v) = \exists w \forall a \forall b ((a = u \wedge b = w) \vee (a = w \wedge b = v)) \Rightarrow \mathsf{reach}_i(a, b).$$

Note that this way, the size of the formula for $reach_{i+1}$ is only a constant size larger then then of $reach_i$. Apply this $\log N = n^c$ times, we get a QBF of size $O(n^c)$ which is true iff $x \in L$. Note that we obviously did a poly-time reduction. A careful examination of the proof shows that the reduction runs in fact in logspace. ■

4 Savitch's Theorem

We prove Savitch's Theorem: $\mathsf{NSPACE}(S(N)) \subset \mathsf{SPACE}(S^2(n))$. We will prove it for $S(n) = O(\log n)$, and it is a simple exercise to see this implies the theorem for all $S(n) \geq \log(n)$.

Theorem 10 (Savich's Theorem [4]) $\mathsf{NL} \subseteq \mathsf{SPACE}(\log^2 n)$. □

Proof Need to show: given a directed graph G on n vertices; s, t are vertices in G; can check if there is a path $s \rightsquigarrow t$ using only $O(\log^2 n)$ space. Let $\mathsf{reach}_i(u, v) \stackrel{\text{def}}{=}$ "there exists a path $u \rightsquigarrow v$ of length $\leq 2^i$". As before, we will use the fact that

$$\mathsf{reach}_{i+1}(u, v) = \exists w \ \mathsf{reach}_i(u, w) \wedge \mathsf{reach}_i(w, v).$$

Note that if we can solve reach_i in space S_i then we can solve reach_{i+1} in space $S_{i+1} = S_i + O(\log n)$, since we need $O(\log n)$ memory to enumerate w, and we can run the two instances of reach_i sequentially using the same space for both. Formally, to solve reach_{i+1} we run the following algorithm:

- run over all $w = 1, \cdots, n$.
  - $\alpha = \mathsf{reach}_i(u, w)$.
  - $\beta = \mathsf{reach}_i(w, v)$.
  - If $\alpha = \beta = 1$, output "Yes".
- Output "No".

The total space we use is $S_{\log n} = O(\log^2 n)$. ■

References

[1] Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009.

[2] Omer Reingold. Undirected st-connectivity in log-space. In *STOC*, pages 376–385, 2005.

[3] Michael E. Saks and Shiyu Zhou. Bp $_{\text{h}}$space(s) subseteq dspace(s$^{3/2}$). *J. Comput. Syst. Sci.*, 58(2):376–403, 1999.

[4] Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177 – 192, 1970.

[5] L. J. Stockmeyer and A. R. Meyer. Word problems requiring exponential time(preliminary report). In *Proceedings of the fifth annual ACM symposium on Theory of computing*, STOC '73, pages 1–9, New York, NY, USA, 1973. ACM.