CSE 200 Computability and Complexity Monday, April 8, 2013

Lecture 3: Time Complexity & Space Complexity

Instructor: Professor Shachar Lovett *Scribe: Dongcai Shen*

1 Intriguing Problems

Matrix Multiplication. Given two $n \times n$ matrices. A trivial algorithm uses $O(n^3)$ time. The apparent lower bound is $\Omega(n^2)$, which is the time to read the matrices. Researchers have discovered a series of algorithms with a lower time complexity. Today's best algorithm is $O(n^{2.3\cdots})$. Some researchers believe that:

- Can achieve $n^{2+\epsilon}$ for arbitrarily small $\epsilon > 0$.

Fast Fourier Transform. A *fast Fourier transform (FFT)* is an algorithm to compute the *discrete Fourier transform (DFT)* and its inverse. Put simply, it's about computing n inverses of n given values. A naïve method consumes $O(n^2)$ arithmetic operations. However, this can be done in $O(n \log n)$ time by the Cooley-Tukey algorithm [2]. However, we still don't know if this is optimal, or if better algorithms exist (say, running in time $O(n)$).

2 Time Complexity

Definition 1 (Time complexity) Let $T : \mathbb{N} \to \mathbb{N}$. A TM M has time complexity T (also $T(n)$) if $\forall x \in \{0,1\}^*$, $M(x)$ halts in at most $T(|x|)$ steps. □

Definition 2 ($\mathsf{TIME}(\cdot)$) $\mathsf{TIME}(T) \stackrel{\text{def}}{=} \{f : \{0,1\}^* \to \{0,1\}^* \text{ s.t. } f \text{ is computable by a TM } M \text{ in time } T\}$ □

Definition 3 (P)

- $\mathsf{P} \stackrel{\text{def}}{=} \cup_{c \geq 1} \mathsf{TIME}(n^c)$. "efficient computation"
- $\mathsf{EXP} \stackrel{\text{def}}{=} \cup_{c \geq 1} \mathsf{TIME}(2^{n^c})$. □

Why do we think poly-time is efficient?

Theorem 4 (Time Hierarchy Theorem (relaxed version)) *Let $T : \mathbb{N} \to \mathbb{N}$ be a computable function, $T(n) \geq n$. $T(n)$ is computable in time $T(n)$. then,* $\mathsf{TIME}(T(n)) \subsetneq \mathsf{TIME}(T^2(n))$. □

Remarks. "$T(n)$ is computable in time $T(n)$" avoids some crazy functions.

Theorem 4 can be refined to the following theorem, which we would not show. Details can be seen in [1] section 1.7.

Theorem 5 (Time Hierarchy Theorem) *Same conditions as in Theorem 4, Then*

$$\mathsf{TIME}(T(n)) \subsetneq \mathsf{TIME}(T(n) \cdot \log T(n)).$$

□

Proof Define $f : \{0,1\}^* \to \{0,1\}$. On input x with $n \stackrel{\text{def}}{=} |x|$.

- Interpret x as describing some TM M, so $x = \langle M \rangle$.

- Simulate running M on input x for $T(n)$ steps.
  - If M halts within $T(n)$ steps, then choose $f(x) \neq M(x)$.
  - If M doesn't halt within $T(n)$ steps, then $f(x) = 0$.

We will show that $f \notin \mathsf{TIME}(T(n))$ but $f \in \mathsf{TIME}(T^2(n))$. The 1st part "$f \notin \mathsf{TIME}(T(n))$" comes from the diagonalization definition of f; the 2nd part "$f \in \mathsf{TIME}(T^2(n))$" comes from the simulation.

Claim 6 $f \notin \mathsf{TIME}(T(n))$. □

Sketch of Proof Say it is, so there is a TM M s.t. $M(x) = f(x)\ \forall x \in \{0,1\}^*$ and M runs in time $T(n)$ but then $f(\langle M \rangle) \neq M(\langle M \rangle)$. Contradiction. ■

Claim 7 $f \in \mathsf{TIME}(T^2(n))$. □

Sketch of Proof Simulate running $M(\langle M \rangle)$ for $T(n)$ steps, where $n \stackrel{\text{def}}{=} |x|$. Simulator:

- Computes $T(n)$.
- Keeps a counter simulates M for at least $T(n)$ steps.

This can be done using the universal machine we defined in time $O(T^2(n))$. ■

■

Concerns. This function is very artificial. What people really care about is some real-world functions like shortest paths. Ever since the 1960s, people have found lots of good space lower bounds, but efforts in figuring out time lower bounds remain in vain.

Corollary 8 $\mathsf{P} \subsetneq \mathsf{EXP}$. □

Question 9 $\mathsf{P} = \mathsf{NP}$ *?* □

3 Space Complexity

Definition 10 (Space complexity) $\mathsf{SPACE}(S(n)) \stackrel{\text{def}}{=} \{$functions which can be computed by a TM which uses space $S(n)\}$. $S(n)$ is the length of the work tape is $S(n)$. □

Caveat. When we talk about the space complexity, we don't count the input tape space. We can view the input as a read-only tape, storing on disk, versus to read/write work tapes.

Definition 11 (Connectivity in graphs)

- Input: undirected graph G with n vertices. Two vertices: s and t.
- Goal: Is there a path from s to t? □

Table 1: Algorithms for s-t connectivity

	TIME	SPACE	**Remark**
BFS	n^2	n	Deterministic
Random walk	n^4	$\log n$	Randomized
Derandomized walk	$n^{O(1)}$	$\log n$	Randomized

Randomized algorithm. Token. on a vertex. Initially, it is on s. At every step, move to a random neighbors. If token on t, answers "Yes".

It turns out that for undirected graphs, in $O(n^3)$ expected time the algorithm returns "Yes".

How to derandomize this problem is a long-awaited open problem. It wasn't proved until 2005 by Omer Reingold [3]. The algorithm looks like a randomized algorithm. Derandomization is also useful for designing efficient data structures. If you are interested, Shachar is going to host a seminar on derandomization in Fall 2013.

Definition 12 (Satisfiability)

- Input: CNF formula: $\phi(x) = (x_1 \vee x_2 \vee x_7) \wedge (x_5 \vee x_8) \wedge \cdots$.
- Goal: Is ϕ satisfiable. □

A naïve solution is just to enumerate all 2^n possible assignments and evaluate each time an assignment is enumerated. The time is $O(2^n)$ and the space is $O(n)$. This shows that space complexity has a different, more crude nature compared to time complexity.

Definition 13 (PSPACE, POLY-LOGSPACE, LOGSPACE)

- $\mathsf{PSPACE} \stackrel{\text{def}}{=} \cup_{c \geq 0} \mathsf{SPACE}(n^c)$.
- $\mathsf{POLY\text{-}LOGSPACE} \stackrel{\text{def}}{=} \cup_{c \geq 0} \mathsf{SPACE}(\log^c n)$.
- $\mathsf{LOGSPACE} \stackrel{\text{def}}{=} \cup_{c \geq 0} \mathsf{SPACE}(c \log n)$. □

Theorem 14 (Space Hierarchy Theorem) *Let $S : \mathbb{N} \to \mathbb{N}$ be such that $S(n) \geq O(\log n)$. Also, let $g : \mathbb{N} \to \mathbb{N}$ be such that $\lim_{n\to\infty} g(n) = \infty$. Then*

$$\mathsf{SPACE}(S(n)) \subsetneq \mathsf{SPACE}(g(S(n)).$$

□

For example, $\mathsf{SPACE}(n) \subsetneq \mathsf{SPACE}(n \log\log\log n)$.

Proof Construct a function $f : \{0,1\}^* \to \{0,1\}$ on an input x. $n \stackrel{\text{def}}{=} |x|$.

- Interpret x as a code of a TM M.
- Run M on x, making sure we use at most $S(n)$ space.
- If at some point, $M(x)$ accesses an index in the work tape which is $> S(n)$, halts and returns 0.
- If not, then choose $f(x) \neq M(x)$.

Simulator f.

- (Tapes) The simulator uses $3S(n) \cdot \log |\Gamma|$ cells to remember the first $S(n)$ symbols in each tape of M.
- (State) State is at most $\log n$. ($|Q|$ states, input length $> |Q|$. state $= \log n$ bits.)
- (Heads) $3 \log S(n)$.
- Simulation will need $O(\log S(n))$ more space.

Simulator uses $O(S(n)) + O(\log n + \log S(n))$ bits, which is less than $g(S(n))$ for large enough n.
■

Proposition 15

- $\mathsf{TIME}(T(n)) \subseteq \mathsf{SPACE}(T(n))$.
- $\mathsf{SPACE}(S(n)) \subseteq \mathsf{TIME}(2^{S(n)})$. □

Proof First part is trivial. The second part follows since with $S(n)$ space there are at most $2^{S(n)}$ different configurations. ■

References

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

[2] James W. Cooley and John W. Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, 19(90):pp. 297–301, 1965.

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