CS 4510: Automata and Complexity **Fall 2019**

Lecture 16: Space and Time Hierarchies

October 30, 2019

Lecturer: Santosh Vempala *Scribe: Aditi Laddha*

16.1 Introduction

16.1.1 Asysmptotic Notations

Definition 16.1 (Big O notation) *Let $f, g : \mathbb{N} \to \mathbb{R}^+$ be functions. Then $g(n) = O(f(n))$ if $\exists n_0 \in \mathbb{N}, c > 0$ such that $\forall n \geq n_0$, $g(n) \leq cf(n)$.*

Intuitively, if $g(n)$ is $O(f(n))$ then $g(n)$ grows no faster than $f(n)$.

Definition 16.2 (Small o notation) *Let $f, g : \mathbb{N} \to \mathbb{R}^+$ be functions. Then $g(n) = o(f(n))$ if $\forall c > 0, \exists n_0 \in \mathbb{N}$ such that $\forall n \geq n_0$, $g(n) \leq cf(n)$.*

Intuitively, if $g(n)$ is $o(f(n))$ then $g(n)$ grows slower than $f(n)$. Some examples demonstrating this,

- $1000\sqrt{n} = O(\frac{n}{100})$, $1000\sqrt{n} = o(\frac{n}{100})$
- $n + 10^7 = O(10n)$
- $n + 10^7 \neq o(10n)$ because for $c = 1/10$, $n + 10^7$
- $\dfrac{n}{\log(n)} = o(n)$
- $n^{0.99} = o(n)$
- $\log(n) = o(\log^2(n))$, $\log^1.99(n) = o(\log^2(n))$
- $n - \sqrt{n} = O(n)$ but $n - \sqrt{n} \neq o(n)$

16.1.2 Space and Time

Definition 16.3 (SPACE) *A language L is said to belong to class $\mathrm{SPACE}(s(n))$ if there exists a TM M that decides L and for input strings of size n, L uses $O(s(n))$ space.*

Note that we do not count input cells in the space used by a TM.

Definition 16.4 (TIME) *A language L is said to belong to class $\mathrm{TIME}(t(n))$ if there exists a TM M that decides L and for input strings of size n, L runs for $O(t(n))$ steps.*

Definition 16.5 (Space-constructible) *A function $s : \mathbb{N} \to \mathbb{N}$ is called space-constructible if $s(n) \geq \log_2(n)$ and there exists a Turing machine that on input 1^n outputs $s(n)$ in binary while using $O(s(n))$ space.*

Definition 16.6 (Time-constructible) *A function $t : \mathbb{N} \to \mathbb{N}$ is called time-constructible if $t(n) \geq n$ and there exists a Turing machine which on input 1^n outputs $t(n)$ in binary while using only $O(t(n))$ steps.*

Definition 16.7 (Configuration of a TM) *The configuration of a Turing machine M can be completely specified by the 3-tuple*

$$\langle \text{Tape Content, State, Head Position} \rangle$$

If we know that a TM M uses at most $s(n)$ space on input of size n, then the number of possible configurations of M while computing on an input of size n is at most $|\Gamma|^{s(n))} \cdot |Q| \cdot (s(n)+n) = 2^{O(s(n))}$ as every cell on the tape can have one of Γ tape symbols and the head can be positioned on the input or any of the $s(n)$ cells used by M.
Question: Does more space or more time give TM more power, i.e., the ability to decide more languages?

16.2 Space Hierarchy Theorem

Theorem 16.8 (Space hierarchy theorem) *For any space-constructible function $s(n)$, there exists a language L that can be decided by a TM using $O(s(n))$ space and cannot be decided by any TM using $o(s(n))$ space. In other words,*

$$\exists L \textit{ such that } L \in \text{SPACE}(O(s(n))) \textit{ and } L \notin \text{SPACE}(o(s(n)))$$

Proof: Let L be the language accepted by the following Turing machine D:
On input $x = (\langle M \rangle, 1^k)$ with $n = |\langle M \rangle, 1^k|$:

1. Compute $s(n)$ using space contructibility and mark $s(n)$ space on the tape
2. Keep a count of the number of steps taken by D
3. If M is not a valid TM description or input is not in the prescribed format, REJECT
4. Run M on input $(\langle M \rangle, 1^k)$
5. If space used exceeds $s(n)$, REJECT
6. If time exceeds $C^{s(n)}$, REJECT where $C \leq |\Gamma| \cdot |Q_M|$
7. If M accepts, REJECT
8. If M rejects, ACCEPT

Keeping a counter for number of steps takes $\log(C^{s(n)}) = O(s(n))$ space. Step (6) ensures that D terminates on all inputs. It is to avoid the case when M does not halt on input x. We claim that the language accepted by D is the required language.

Claim 16.9 *L can be decided using $O(s(n))$ space.*

D decides L and uses $O(s(n))$ space and always terminates.

Claim 16.10 *No TM using $o(s(n))$ space can decide L.*

Suppose there is a TM, M_L that decides L using $g(n) = o(s(n))$ space. Since M_L uses $o(s(n))$ space, there exists some large enough k such that $g(|\langle M_L \rangle, 1^k|) \leq s(|\langle M_L \rangle, 1^k|)$ and M_L uses less than $s(n)$ space. Run D on $(\langle M_L \rangle, 1^k)$. Since M_L uses $o(s(n))$ space D can simulate it and will not stop at step (5) or (6). If M_L accepts $(\langle M_L \rangle, 1^k)$, D rejects it and if M_L rejects $(\langle M_L \rangle, 1^k)$, D accepts it. D and M_L decide the same language but give different output on input $(\langle M_L \rangle, 1^k)$, a contradiction. ■

We don't know whether $\exists L$ such that $L \in \text{TIME}(O(s(n))$ and$L \notin \text{TIME}(o(s(n)))$? Why does the same proof not work for time? Where should we keep the timer? Space is not bounded. Keep the timer with you logtn overhead

Theorem 16.11 (Time hierarchy theorem) *For any time-constructible function $s(n)$, there exists a language L that can be decided by a TM using $O(t(n))$ time and cannot be decided by any TM using $o\left(\frac{t(n)}{\log(t(n))}\right)$ time. In other words,*

$$\exists L \textit{ such that } L \in \text{TIME}(O(t(n))) \textit{ and } L \notin \text{TIME}\left(o\left(\frac{t(n)}{\log(t(n))}\right)\right)$$

For proof, see Theorem 9.10 in "Introduction to the Theory of Computation".

16.3 References

- Ch 7.1 Measuring Complexity, "Introduction to the Theory of Computation"
- Ch 9.1 Hierarchy Theorems, "Introduction to the Theory of Computation"