CS 170 Efficient Algorithms and Intractable Problems **Section Notes**

Jan 24, 2018 Summary

TA: Chinmay Nirkhe

1 Introductions

1.1 Me

1. Chinmay Nirkhe: First year PhD student studying quantum computation. My interests are in the quantum PCP conjecture, multiprove interactive proofs, and quantum supremacy. Come talk to me about these problems!
2. Contact Information:
   (a) `nirkhe@cs.berkeley.edu`.
   (b) 615 Soda Hall
   (c) `http://cs.berkeley.edu/~nirkhe`
3. Resources:
   (a) I will post any documents I generate, specific to my sections at

   `http://cs.berkeley.edu/~nirkhe/cs170notes`

   (b) I used to TA a similar course at Caltech. My course notes (which include a lot of similar problems) and section notes from that are also on my website.
   You can find them at `http://cs.berkeley.edu/~nirkhe/cs38notes`.
4. My Schedule: I teach 2 sections; they are at Wednesday 9am and 10am. I have my office hours on Wednesday 12pm - 2pm. Generally, you can find me in my office but I might sometimes turn you away if I am busy.

1.2 You

I want to get to know you. Please fill out an index card with the following information:

1. Name
2. Email
3. Major (Declared or Intended) and Graduation Year
4. Favorite problem in Computer Science or Electrical Engineering
5. Why do you want to learn about Algorithms?
6. Any concerns or questions I can answer.

2 Review of Big-O Notation

Topics Covered: Partial Ordering, Asymptotic Definition, Thinking of $O(f)$ as a set.

Problem 1 *Show that the following two definitions are equivalent.*

1. *For functions $f, g : \mathbb{N} \to \mathbb{N}$, $f = O(g)$ if there exists a $c > 0, N > 0$, s.t. $\forall n \geq N$, $f(n) \leq c \cdot g(n)$.*
2. *For functions $f, g : \mathbb{N} \to \mathbb{N}$, $f = O(g)$ if there exists a $c > 0$ s.t. $f(n) \leq c \cdot g(n)$ for all but finitely many n.*

3 The Master Theorem

Theorem 2 (Master Theorem) *If $T(n)$ satisfies the above recurrence relation, then*

- *if $\exists\, c > 1, n_0$ such that for $n > n_0$, $af(n/b) \geq cf(n)$ then $T(n) \in \Theta(n^{\log_b a})$*
- *if $f(n) \in \Theta(n^{\log_b a})$ then $T(n) \in \Theta(n^{\log_b a} \log n)$*
- *if $\exists\, c < 1, n_0$ such that for $n > n_0$, $af(n/b) \leq cf(n)$ then $T(n) \in \Theta(f(n))$*

Proof: Convince yourself that $a^{\log_b n} = n^{\log_b a}$. By induction, its easy to see that

$$T(n) = a^j \cdot T\left(\frac{n}{b^j}\right) + \sum_{k=0}^{j} a^k f\left(\frac{n}{b^k}\right) \tag{1}$$

Apply to $j = \log_b n$ and recognize $T(1)$ is a constant so

$$T(n) = \Theta(a^{\log_b n}) + \sum_{k=0}^{\log_b n} a^k f\left(\frac{n}{b^k}\right) \tag{2}$$

Let's consider the first case. We apply the relation to get[1]

$$\begin{aligned} T(n) &= \Theta(a^{\log_b n}) + a^{\log_b n} f(1) \sum_{k=0}^{\log_b n} c^{-k} \\ &= \Theta(a^{\log_b n}) + \Theta\left(a^{\log_b n} \frac{c}{c-1}\right) \\ &= \Theta(a^{\log_b n}) = \Theta(n^{\log_b a}) \end{aligned} \tag{3}$$

For the second case,

$$\begin{aligned} T(n) &= \Theta(a^{\log_b n}) + \sum_{k=0}^{\log_b n} a^k \Theta\left(\left(\frac{n}{b^k}\right)^{\log_b a}\right) \\ &= \Theta(n^{\log_b a}) + \sum_{k=0}^{\log_b n} a^k \left(\frac{\Theta(n^{\log_b a})}{a^k}\right) \\ &= \Theta(n^{\log_b a} \log n) \end{aligned} \tag{4}$$

For the third case, recognize that it's nearly identical to the first except the summations are bounded in the other direction which leaves $f(n)$ as the dominating term. $\square$

4 Problems from general TA Section Notes

[1] I've made the simplifaction here that $n_0 = 0$. As an excercise, convince yourself this doesn't effect anything, just makes the algebra a little more complicated.