Computer Science 311
Recitation 3

Weisi Fan

Alvin Chon

02-15-2020

1 Review Notes

Please review the methods of solving recurrences

1. **Master Theorem**
   Let $a > 1$ and $b > 1$ be constants, let $f(n)$ be a function, and let $T(n)$ be defined on the nonnegative integers by the recurrence

   $$T(n) = aT(n/b) + f(n)$$

   a) If $f(n) = O(n^{\log_b a-\epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$
   b) If $f(n) = O(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a}\log n)$
   c) If $f(n) = O(n^{\log_b a+\epsilon})$ for some constant $\epsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

2. **Substitution**
   Guess a bound and then use mathematical induction to prove that guess is correct. Remember that you must solve the exact form of your guess.

3. **Recursion Tree**
   Convert the recurrence into a tree whose nodes represent the costs incurred at various levels of the recursion. Then add up the work over all nodes in the tree.

2 Master Theorem Practice

4 practice problems, 1 for each case of the Master Theorem. Decide which problem is which case (a, b, or c) from the Review Notes section.
So why are there 4 problems? It helps to write out what a, b, and f(n) are. Take about 6 mins and then we'll discuss them. Solutions are at the end of the document.

1. $T(n) = T(\frac{n}{2}) + 2^n$
2. $T(n) = 3T(\frac{n}{2}) + n$
3. $T(n) = 64T(\frac{n}{8}) - nlgn$
4. $T(n) = 4T(\frac{n}{2}) + n^2$

3 Solving Recurrences

3.1 False Proof: Pitfall

$T(n) = 2T(\frac{n}{2}) + n$
We know this is $O(nlgn)$ where the base of the log is 2.
Prove $T(n) = O(n)$ by guessing $T(n) \le cn$.
Argue $T(n) \le 2(\frac{cn}{2}) + n = cn + n = O(n)$. This works right?
Unfortunately we had to prove $T(n) = O(n)$ and not $T(n) = (c+1)n$.
They are not the same thing.

3.2 Changing Variables

Changing variables

Consider $T(n)$ defined by the recurrence
$T(n) = 2T(2^{\lfloor (\lg n)/2 \rfloor}) + \lg n$.

We simplify the recurrence by changing the variable $n = 2^m$
$T(2^m) = 2T(2^{\lfloor m/2 \rfloor}) + m$.

By defining $S(m) = T(2^m)$, we obtain the new recurrence
$S(m) = 2S(\lfloor m/2 \rfloor) + m$.

It can be shown by mathematical induction that
$S(m)$ is in $O(m \lg m)$.

Changing back from $S(m)$ to $T(n)$, we obtain
$T(n) = T(2^m) = S(m) = O(m \lg m) = O(\lg n \lg \lg n)$.

3.3 Solve the following recurrences and indicate the running time of each $T(n)$.

1. $T(n) = T(n/3) + n$; $T(1) = 1$.
2. $T(n) = T(n-2) + n$; $T(n) = 1$ for $n \le 2$.
3. $T(n) = 2T(n/2 + 2) + n$; $T(n) = 1$ for $n \le 6$.
4. $T(n) = 4T(n/4) + 3\log_4 n$; $T(n) = 1$ for $n \le 4$.
5. $T(n) = T(n-2) + 4\log n$; $T(n) = 1$ for $n \le 2$.
6. $T(n) = T(\alpha n) + T((1-\alpha)n)$; $T(1) = 1$ and $0 < \alpha < 1$.

4 Solutions: Master Theorem Practice

$T(n) = T(\frac{n}{2}) + 2^n$
$a = 1, b = 2, f(n) = 2^n, n^{log_2 1} = 1$
Third case of the Master Theorem. Also, $af(\frac{n}{b}) = 2^{\frac{n}{2}} \leq 2^n = f(n)$ for all $n > 0$.
$T(n) = \Theta(2^n)$

$T(n) = 3T(\frac{n}{2}) + n$
$a = 3, b = 2, f(n) = n, n^{log_2 3} = n^{log_2 3}$
First case of the Master Theorem. $T(n) = \Theta(n^{log_2 3})$

$T(n) = 64T(\frac{n}{8}) - nlgn$
$a = 64, b = 8, f(n) = -nlgn, n^{log_8 64} = n^2$
Master Theorem does not apply since $f(n)$ is not positive.

$T(n) = 4T(\frac{n}{2}) + n^2$
$a = 4, b = 2, f(n) = n^2, n^{log_2 4} = n^2$
Second case of the Master Theorem. $T(n) = \Theta(n^2 logn)$