

Analysis of Algorithms CS 3343 Lecture Six

Prof. William Winsborough
September 23, 2008

Business

- Read section 4.3, 4.4.1
- Recall: Homework 3 Due Thursday September 25:
 - Exercise: 4.2-1
- Homework 4 Due Tuesday September 30
 - Problem 4.1 (p.85)
- No class Thursday Oct. 2
 - I'm giving a keynote talk at PST2008, Sixth Annual Conference on Privacy, Security and Trust, Fredericton, New Brunswick, Canada
- Tuesday October 7 will be devoted to review
 - Bring your questions
- Thursday October 9 will be midterm 1

23 September 2008

Winsborough CS 3343 Lecture 6

2

The Recursion-tree Method

- Nodes represent the cost of a single subproblem among the recursive function invocations
- Sum up the costs within each level of the tree, then sum over the levels
- Basic method:
 - Allow a little sloppiness
 - Use the approach to come up with a good guess at a solution
 - Verify the solution is correct by using the substitution method

23 September 2008

Winsborough CS 3343 Lecture 6

3

Example 1

- $T(n) = 3T(\lfloor n/4 \rfloor) + \Theta(n^2)$
 - Ignore floor and assume $n = 4^k$ for some k
- Create tree for $T(n) = 3T(n/4) + cn^2$
- The size of the subproblem at level i is $n/4^i$
 - Reach the base case when $n/4^i = 1$, or $i = \log_4 n$
 - So height of tree is $\log_4 n + 1$
- Number of nodes at depth i is 3^i and each node has cost $c(n/4^i)^2$
 - Total cost of level i is $3^i c(n/4^i)^2 = (3/16)^i cn^2$
- The bottom level has $3^{\log_4 n} = n^{\log_4 3}$ nodes, each contributing cost $\Theta(1)$, for a total cost of $\Theta(n^{\log_4 3})$

23 September 2008

Winsborough CS 3343 Lecture 6

4

Summing the Levels

- Refer to p68-70 in text

23 September 2008

Winsborough CS 3343 Lecture 6

5

Example 2

- $T(n) = T(n/3) + T(2n/3) + \Theta(n)$
- The longest path is the one that follows $T(2n/3)$ at each level
 - $(2/3)^k n = 1$ when $k = \log_{3/2} n$, so this is the height of the tree
- The sum of the costs at each level is at most cn
- If the tree were a complete binary tree,
 - The sum of all levels would be $cn \log_{3/2} n = O(n \lg n)$
 - There would be $2^{\log_{3/2} n} = n^{\log_{3/2} 2}$ leaves for a total cost of $\Theta(n^{\log_{3/2} 2})$
 - This is asymptotically larger than $n \lg n$
- But the tree is far from complete, so let's guess the cost is dominated by the internal nodes, not the leaves
 - Guess: $T(n) = O(n \lg n)$
 - Apply substitution method to verify

23 September 2008

Winsborough CS 3343 Lecture 6

6

Master Method

- Master theorem

The Master Theorem applies to recurrences of the following form:

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

where $a \geq 1$ and $b > 1$ are constants and $f(n)$ is an asymptotically positive function.

There are 3 cases:

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a} \log^k n)$ with $k \geq 0$, then $T(n) = \Theta(n^{\log_b a} \log^{k+1} n)$.
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ with $\epsilon > 0$, and $f(n)$ satisfies the regularity condition, then $T(n) = \Theta(f(n))$.
Regularity condition: $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n .