CS 381 HW 1

Ankur Dhoot

1.

a) T(n) = 9T(n/3) + n^2 for $n > 2$ and T(n) = 1 otherwise

a = 9 b = 3 f(n) = n^2

Since $n^{\log_b a} = n^{\log_3 9} = n^2$ and f(n) = $\Theta(n^2)$, f(n) = $\Theta(n^{\log_b a})$. Thus, we are in case 2 of the master theorem. Therefore, T(n) = $\Theta(n^2 \lg n)$.

b) T(n) = 6T(n/2) + $n^{2.4}$ for $n > 1$ and T(n) = 1 otherwise

a = 6 b = 3 f(n) = $n^{2.4}$

Since $n^{\log_b a} = n^{\log_2 6} = n^{2.58}$ and f(n) = $\Theta(n^{2.4})$, f(n) = $O(n^{2.58-\epsilon})$ for $0 < \epsilon < .18$. Thus, we are in case 1 of the master theorem. Therefore, T(n) = $\Theta(n^{\log_2 6}) = \Theta(n^{2.58})$.

c) T(n) = 12T(n/4) + n^2 for $n > 3$ and T(n) = 1 otherwise

a = 12 b = 4 f(n) = n^2

Since $n^{\log_b a} = n^{\log_4 12} = n^{1.79}$ and f(n) = $\Theta(n^2)$, f(n) = $\Omega(n^{1.79+\epsilon})$ for $0 < \epsilon < .20$. Thus, we are in case 3 of the master theorem. Therefore, T(n) = $\Theta(f(n)) = \Theta(n^2)$.

2. Proof: By induction on n

Base case: n = {1,2,3}

For all n $\in$ {1,2,3}, T(n) = 1 $\leq$ 12n. Thus, the statement holds for the base case.

Induction Step:

Inductive Hypothesis: Suppose that the statement holds for all positive $m < n$. That is T(m) $\leq$ 12m.

Now, we must show that the statement holds for n. That is, we must show that T(n) $\leq$ 12n.

T(n) = T($\lfloor 2n/3 \rfloor$) + T($\lfloor n/4 \rfloor$) + n

$\leq 12\lfloor 2n/3 \rfloor + 12\lfloor n/4 \rfloor + n$ (by induction hypothesis)

$\leq 12(2n/3) + 12(n/4) + n$ (by definition of floor function)

$= 8n + 3n + n$

=12n.

Thus, having established the basis and induction step, the claim follows by the principle of mathematical induction.

3. T(n) = 1 for $n \leq 2$

T(n) = T($\lfloor 3n/5 \rfloor$) + T($\lfloor 2n/5 \rfloor$) + cn for $n > 2$

Recursion Tree argument:

The longest simple path from root to a leaf is $n \to \lfloor 3n/5 \rfloor \to \lfloor 3\lfloor 3n/5 \rfloor/5 \rfloor \to$ ... $\to$ 1 Thus, the height of the tree is at most $\log_{5/3} n$. Consider the first level of the tree. The size of the "subproblem" is $\lfloor 3n/5 \rfloor$ and $\lfloor 2n/5 \rfloor$

which will add cost $\mathrm{c}\lfloor 3n/5 \rfloor + \mathrm{c}\lfloor 2n/5 \rfloor \leq cn$. The second level will have 4 "subproblems" of size $\lfloor 3\lfloor 3n/5 \rfloor/5 \rfloor$, $\lfloor 2\lfloor 3n/5 \rfloor/5 \rfloor$, $\lfloor 3\lfloor 2n/5 \rfloor/5 \rfloor$, $\lfloor 2\lfloor 2n/5 \rfloor/5 \rfloor$, which will contribute cost $\mathrm{c}\lfloor 3\lfloor 3n/5 \rfloor/5 \rfloor + c\lfloor 2\lfloor 3n/5 \rfloor/5 \rfloor + c\lfloor 3\lfloor 2n/5 \rfloor/5 \rfloor + c\lfloor 2\lfloor 2n/5 \rfloor/5 \rfloor \leq c(9n/25) + c(6n/25) + c(6n/25) + c(4n/25) = cn$. It's easy to see that every level of the tree will contribute at most cn to the cost of the tree with the cost per level decreasing as we move further down the tree since the tree isn't a complete binary tree and more internal nodes will be missing. Since the total cost is the sum of the cost at each level which is bounded by cn and there are at most $\log_{5/3} n$ levels, the total cost $T(n)$ is $O(cn * \log_{5/3} n) = O(n \lg n)$.

Sketch of first two levels of recursion tree: