Quiz 4/5

Nathan Bickel
CSCE 350 - Data Structures and Algorithms

April 13, 2023

Master Theorem 1 *Let $a \geq 1$ and $b > 1$ be integers, and $T(n) = aT(n/b) + f(n)$ be a recurrence relation. If $f(n) \in \Theta(n^d)$ where $d \geq 0$, then*

$$T(n) \in \begin{cases} \Theta(n^d) & \text{if } a < b^d \\ \Theta(n^d \log n) & \text{if } a = b^d \\ \Theta(n^{\log_b a}) & \text{if } a > b^d \end{cases}.$$

1. We will consider comparison to be the basic operation. Let $C(n)$ be the number of comparisons made when A is an array of length n. Each time the algorithm is run, 5 comparisons are made, and the algorithm is run on the child of the node at vertex i (if there is one). Since a heap is a binary tree, this sub-heap will be about half the size. So we have

$$C(n) = C(n/2) + 5, C(1) = 5.$$

So $C(n) = aT(n/b) + f(n)$, where $a = 1, b = 2, f(n) = 5$. Since $f(n) \in \Theta(n^d)$ for $d = 0$, and thus $1 = a = b^d = 2^0 = 1$, we have from the Master Theorem that

$$C(n) \in \Theta(n^d \log n) = \Theta(\log n).$$

2. We will consider exponentiation to be the basic operation. Let $P(n)$ be the number of exponentiations made for n inputs. Since the method is recursively called for two lists with size $n/2$, and $n/2$ exponentiations are made afterward, we have

$$P(n) = 2P(n/2) + n/2, P(0) = 0.$$

So $P(n) = aT(n/b) + f(n)$, where $a = 2, b = 2, f(n) = n/2$. Since $f(n) \in \Theta(n^d)$ for $d = 1$, and thus $2 = a = b^d = 2^1 = 2$, we have from the Master Theorem that

$$P(n) \in \Theta(n^d \log n) = \Theta(n \log n).$$

3. We will consider subtraction to be the basic operation. Let $S(n)$ be the number of subtractions made when P is an array of length n. The middle two elements (if there are two) will be subtracted when the method is called, and the method is also recursively called for two lists with size about $n/2$. So we have

$$S(n) = 2S(n/2) + 1, S(0) = 0.$$

So $S(n) = aT(n/b) + f(n)$, where $a = 2, b = 2, f(n) = 1$. Since $f(n) \in \Theta(n^d)$ for $d = 0$, and thus $2 = a > b^d = 2^0 = 1$, we have from the Master Theorem that

$$P(n) \in \Theta(n^{\log_b a}) = \Theta(n).$$

4. We will consider addition to be the basic operation. Let $A(n)$ be the number of additions made when A is a square matrix with n rows. We recursively run the algorithm 8 times, where the first input has $n/2$ rows, and we add 4 pairs of these results. So we have

$$A(n) = 8A(n/2) + 4, A(1) = 0.$$

So $A(n) = aA(n/b) + f(n)$, where $a = 8, b = 2, f(n) = 1$. Since $f(n) \in \Theta(n^d)$ for $d = 0$, and thus $8 = a > b^d = 2^0 = 1$, we have from the Master Theorem that

$$A(n) \in \Theta(n^{\log_b a}) = \Theta(n^{\log_2 8}) = \Theta(n^3).$$

5. We will consider comparison to be the basic operation. Let $C(n)$ be the number of comparisons made when with size n. The algorithm will find the defect using 12 comparisons (three in each quadrant: one to check the size and two to check the defectRow and defectColumn), and then run the tiling algorithm in each quadrant. Regardless of whether or not the quadrant is defective, the algorithm will be called with size $n/2$. So we have

$$C(n) = 4C(n/2) + 12, C(1) = 1.$$

So $C(n) = aC(n/b) + f(n)$, where $a = 4, b = 2, f(n) = 12$. Since $f(n) \in \Theta(n^d)$ for $d = 0$, and thus $4 = a > b^d = 2^0 = 1$, we have from the Master Theorem that

$$C(n) \in \Theta(n^{\log_b a}) = \Theta(n^{\log_2 4}) = \Theta(n^2).$$