CS 161 Winter 2021 Section 2

January 19 2021

Recurrence Relations

Recall the Master theorem from lecture:

Theorem 0.1 *Given a recurrence* $T(n) = aT(\frac{n}{b}) + O(n^d)$ *with* $a \geq 1$*, and* $b > 1$ *and* $T(1) = \Theta(1)$*, then*

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

What is the Big-Oh runtime for algorithms with the following recurrence relations?

1. $T(n) = 3T(\frac{n}{2}) + \Theta(n^2)$
2. $T(n) = 4T(\frac{n}{2}) + \Theta(n)$
3. $T(n) = 2T(\sqrt{n}) + O(\log n)$

Divide and Conquer

You have now seen how digit multiplication can be improved upon with divide and conquer. Let us see a more generalized example of Matrix-vector multiplication. Assume that we have $n \times n$ matrix X and $n \times 1$ vector y and we'd like to multiply them.

1. What is the naive solution and what is its runtime?
2. Now let's divide up the problem into smaller chunks like this, where the four $\frac{n}{2} \times \frac{n}{2}$ sub-matrices (A, B, C, D) are each quarters of the original matrices, X and the two $\frac{n}{2} \times 1$ sub vectors, e, g are the split of y:

$$Xy = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} e \\ g \end{bmatrix} = \begin{bmatrix} Ae + Bg \\ Ce + Dg \end{bmatrix}$$

   We now have a divide and conquer strategy! Find the recurrence relation of this strategy and the runtime of this algorithm.

3. Bonus: Can we do better?