CS 341 - Algorithms	**Winter 2018**

Lecture 4: January 16, 2018

Lecturer: Bin Ma — *Notes By: Harsh Mistry*

4.1 Recursion Continued

4.1.1 Master Theorem

$$T(n) = \begin{cases} \Theta(n^c), \text{if } c > \log_b a \\ \Theta(n^c \cdot \log n), \text{ if } c = \log_b a \\ \Theta(n^{\log_b a}), \text{if } c < \log_b a \end{cases}$$

Proof: Proof for case 1.

Base Case : Base case is when n is a small constant where the theorem is obviously true

I.H. : $T(n) \leq \gamma \cdot n^c$

I.S :

$$\begin{aligned} T(n) &= a \cdot T(\frac{n}{b}) + n^c \\ &\leq a \cdot \gamma \frac{n}{b}^x + n^c \\ &= (\gamma \cdot a \cdot b^{-c} + 1) \cdot n^c \end{aligned}$$

Let $\gamma = \frac{1}{1 - \frac{n}{b^c}}$, it is esy to check that

$$(\gamma \cdot a \cdot b^{-c} + 1) \cdot n^c \leq \gamma n^c$$

■

4.1.1.1 Modify Induction conclusion

$T(n) \leq 2T(\frac{n}{2}) + \sqrt{n}$. We guess $T(n) = O(n)$. In induction, we have $T(n) \leq 2c \cdot \frac{n}{2} + \sqrt{n} \leq c \cdot n + \sqrt{n}$. This does not suffice to prove $T(n) \leq c \cdot n$.

This can be solved by proving a slightly modified property. We want to introduce some $-\sqrt{n}$ in $T(\frac{n}{2})$ in order to cancel out the $\sqrt{n}$. Specifically, we will prove $T(n) \leq c \cdot n - 3\sqrt{n}$ for a sightly larger c.

This does not change our conclusion. However, during induction, we have

$$T(n) \leq 2T(\frac{n}{2}) + \sqrt{n} \leq 2\left(c \cdot \frac{n}{2} - 3\sqrt{\frac{n}{2}}\right) + \sqrt{n} \leq cn - \left(\frac{6}{\sqrt{2}} - 1\right)\sqrt{n} \leq c \cdot n - 3\sqrt{n}$$

4.1.1.2 Variable Substitution

To solve $T(n) = 2T(\sqrt{n}) + \log_2 n$.

Let $S(m) = T(2^{m/2}) + m = 2S(\frac{m}{2} + m)$, then $S(m) = m \log m$. Therefore, $T(n) = S(\log_2 n) = \log_2 n \cdot log_2 n \cdot \log_2 \log_2 n)2$.

4.2 Divide and Conquer

The divide and conquer portion of class was all example. Refer to course slides or notes for additional examples