Recursion-tree and Master Methods

Kuan-Yu Chen (陳冠宇)

2019/03/13 @ TR-310-1, NTUST

Review

- There are three methods for obtaining asymptotic "Θ" or "O" bounds on the solution
  - Substitution method
    1. Guess the form of the solution
    2. Use mathematical induction to find the constants and show that the solution works
  - Recursion-tree method
  - Master method

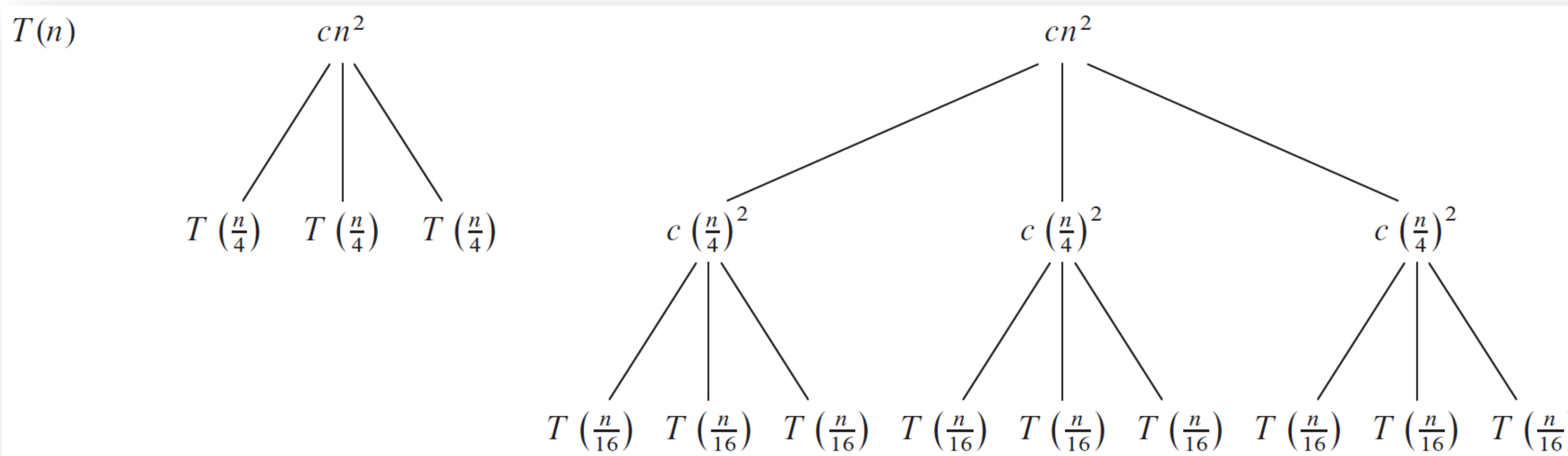
Recursion-tree Method.

- Although you can use the substitution method to provide a solution to a recurrence, you might have trouble coming up with a good guess

- Recursion-tree method serves as a straightforward way to devise a good guess
  - In a ***recursion tree***, each node represents the cost of a single subproblem
  - We sum all of the costs to determine the total cost of the recursion

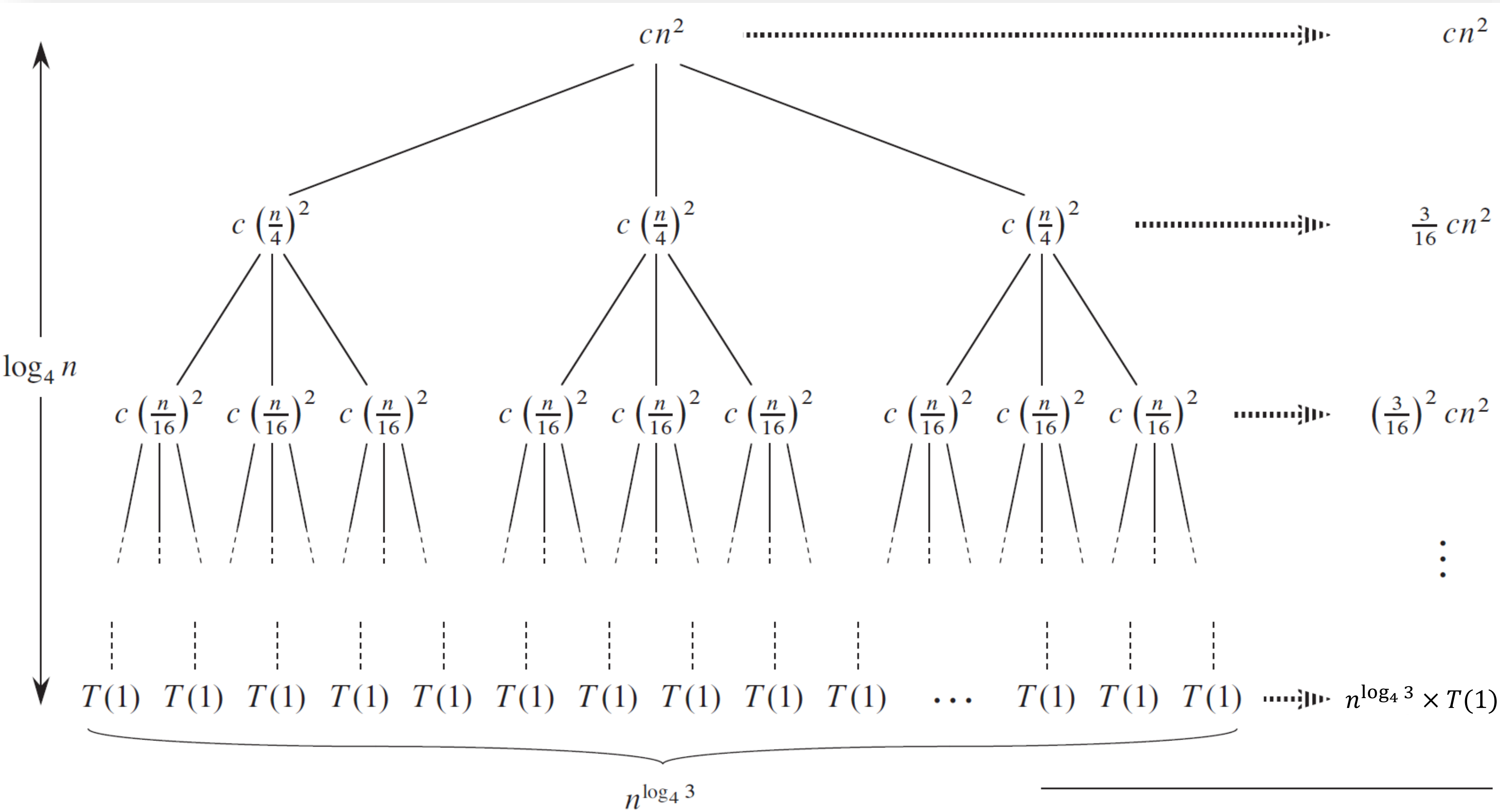
Recursion-tree Method..

- Take $T(n) = 3T\left(\frac{n}{4}\right) + cn^2$ for example
  - For convenience, we assume that n is an exact power of 4



Recursion-tree Method...

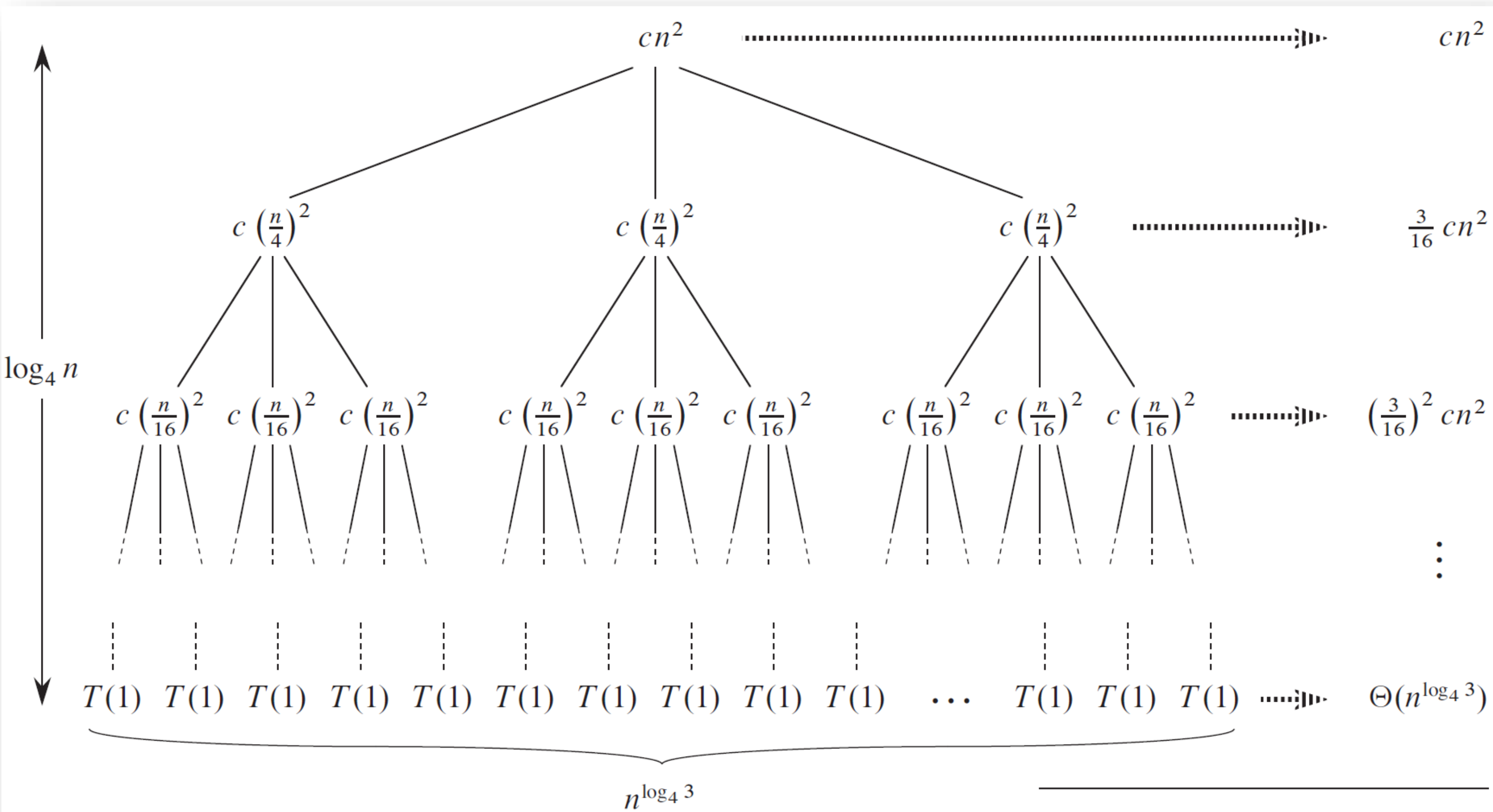
– The fully expanded tree has height $\log_4 n$, that is it has $(\log_4 n) + 1$ levels



Recursion-tree Method....

– Now we add up the costs over all levels to determine the cost for the entire tree

$$T(n) = cn^2 + \frac{3}{16}cn^2 + \left(\frac{3}{16}\right)^2 cn^2 + \cdots + \left(\frac{3}{16}\right)^{\log_4 n - 1} cn^2 + n^{\log_4 3}T(1)$$



Recursion-tree Method.....

– We assume that $T(1)$ is a constant

$$T(n) = cn^2 + \frac{3}{16}cn^2 + \left(\frac{3}{16}\right)^2 cn^2 + \cdots + \left(\frac{3}{16}\right)^{\log_4 n-1} cn^2 + n^{\log_4 3}T(1)$$

$$= cn^2 + \frac{3}{16}cn^2 + \left(\frac{3}{16}\right)^2 cn^2 + \cdots + \left(\frac{3}{16}\right)^{\log_4 n-1} cn^2 + \Theta\left(n^{\log_4 3}\right)$$

$$= \Theta\left(n^{\log_4 3}\right) + cn^2\left(\sum_{i=0}^{\log_4 n-1}\left(\frac{3}{16}\right)^i\right)$$

$$= \Theta\left(n^{\log_4 3}\right) + cn^2\frac{\left(\frac{3}{16}\right)^{\log_4 n} - 1}{\frac{3}{16} - 1}$$

- Too hard to analysis!!

Recursion-tree Method……

– Let's back up one step

$$T(n) = cn^2 + \frac{3}{16}cn^2 + \left(\frac{3}{16}\right)^2 cn^2 + \cdots + \left(\frac{3}{16}\right)^{\log_4 n - 1} cn^2 + n^{\log_4 3}T(1)$$

$$= \Theta\left(n^{\log_4 3}\right) + cn^2\left(\sum_{i=0}^{\log_4 n - 1}\left(\frac{3}{16}\right)^i\right)$$

$$< \Theta\left(n^{\log_4 3}\right) + cn^2\left(\sum_{i=0}^{\infty}\left(\frac{3}{16}\right)^i\right)$$

$$= \Theta\left(n^{\log_4 3}\right) + cn^2\frac{1}{1 - \frac{3}{16}}$$

$$= \Theta\left(n^{\log_4 3}\right) + cn^2\frac{16}{13}$$

$$= O(n^2)$$

Master Method.

- The master method provides a "cookbook" method for solving recurrences of the form

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

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

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 $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log_2 n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(\frac{n}{b}) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

Master Method..

- The master method theorem

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

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 $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log_2 n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(\frac{n}{b}) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

– Take $T(n) = 9T\left(\frac{n}{3}\right) + n$ for example

- $a = 9, b = 3,$ and $f(n) = n$
- $f(n) = n = O\left(n^{\log_b a - \epsilon}\right) = O\left(n^{\log_3 9 - \epsilon}\right)$, where $\epsilon = 1$
- $T(n) = \Theta\left(n^{\log_b a}\right) = \Theta\left(n^{\log_3 9}\right) = \Theta(n^2)$
- Case1

Master Method...

- The master method theorem

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

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 $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log_2 n)$
3. If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if $af(\frac{n}{b}) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

– Take $T(n) = T\left(\frac{2n}{3}\right) + 1$ for example

- $a = 1, b = \frac{3}{2}$, and $f(n) = 1 \Longrightarrow n^{\log_b a} = n^{\log_{\frac{3}{2}} 1} = n^0 = 1$
- $f(n) = 1 = \Theta(1) = \Theta\left(n^{\log_b a}\right)$
- $T(n) = \Theta\left(n^{\log_b a} \log_2 n\right) = \Theta(\log_2 n)$
- Case2

Master Method….

- The master method theorem

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

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 $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a}\log_2 n)$
3. If $f(n) = \Omega(n^{\log_b a+\epsilon})$ for some constant $\epsilon > 0$, and if $af\left(\frac{n}{b}\right) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

– Take $T(n) = 3T\left(\frac{n}{4}\right) + n\log_2 n$ for example

- $a = 3, b = 4$, and $f(n) = n\log_2 n \Longrightarrow n^{\log_b a} = n^{\log_4 3}$
- $f(n) = \Omega(n^{\log_4 3+\epsilon})$
- $af\left(\frac{n}{b}\right) = 3f\left(\frac{n}{4}\right) = 3\frac{n}{4}\log_2\frac{n}{4} = \frac{3}{4}n\log_2 n - \frac{3}{2}n \leq cn\log_2 n = cf(n)$, when $c = \frac{3}{4}$
- $T(n) = \Theta\big(f(n)\big) = \Theta(n\log_2 n)$
- Case3

Master Method.....

- The master method theorem

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

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 $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log_2 n)$
3. If $f(n) = \Omega(n^{\log_b a+\epsilon})$ for some constant $\epsilon > 0$, and if $af(\frac{n}{b}) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$

- In each of the three cases, we compare $f(n)$ with $n^{\log_b a}$
- In case1, $n^{\log_b a}$ is larger than $f(n)$, thus $T(n) = \Theta(n^{\log_b a})$
- In case3, $f(n)$ is larger than $n^{\log_b a}$, thus $T(n) = \Theta(f(n))$
- In case2, $f(n)$ and $n^{\log_b a}$ are the same size, we multiply by a logarithmic factor, and the solution is $T(n) = \Theta(n^{\log_b a} \log_2 n) = \Theta(f(n) \log_2 n)$

Questions?

kychen@mail.ntust.edu.tw