طراحی و تحلیل الگوریتم ها

دکتر امیر لکی زاده

استادیار گروه مهندسی کامپیوتر دانشگاه قم

Master Theorem : قضیه اصلی

- Given: a *divide and conquer* algorithm
- An algorithm that divides the problem of size n into a sub problems, each of size n/b
- Let the cost of each stage (i.e., the work to divide the problem + combine solved sub problems) be described by the function $f(n)$
- Then, the **Master Theorem** gives us a cookbook for the algorithm's running time:

Master Theorem : قضیه اصلی

- if T(n) = aT(n/b) + f(n) then

$$T(n) = \begin{cases} \Theta\left(n^{\log_b a}\right) & f(n) = O\left(n^{\log_b a - \varepsilon}\right) \\ \Theta\left(n^{\log_b a} \log n\right) & f(n) = \Theta\left(n^{\log_b a}\right) \\ \Theta\left(f(n)\right) & f(n) = \Omega\left(n^{\log_b a + \varepsilon}\right) \text{ AND} \\ & af(n/b) < cf(n) \ \text{ for large } n \end{cases} \quad \begin{matrix} \varepsilon > 0 \\ c < 1 \end{matrix}$$

Master Theorem : قضیه اصلی

$$T(n) = 9T(n/3) + n .$$

For this recurrence, we have $a = 9$, $b = 3$, $f(n) = n$, and thus we have that $n^{\log_b a} = n^{\log_3 9} = \Theta(n^2)$. Since $f(n) = O(n^{\log_3 9 - \epsilon})$, where $\epsilon = 1$, we can apply case 1 of the master theorem and conclude that the solution is $T(n) = \Theta(n^2)$.

Master Theorem : قضیه اصلی

$T(n) = T(2n/3) + 1,$

in which $a = 1$, $b = 3/2$, $f(n) = 1$, and $n^{\log_b a} = n^{\log_{3/2} 1} = n^0 = 1$. Case 2 applies, since $f(n) = \Theta(n^{\log_b a}) = \Theta(1)$, and thus the solution to the recurrence is $T(n) = \Theta(\lg n)$.

Master Theorem : قضیه اصلی

$$T(n) = 3T(n/4) + n \lg n \ ,$$

we have $a = 3$, $b = 4$, $f(n) = n \lg n$, and $n^{\log_b a} = n^{\log_4 3} = O(n^{0.793})$. Since $f(n) = \Omega(n^{\log_4 3 + \epsilon})$, where $\epsilon \approx 0.2$, case 3 applies if we can show that the regularity condition holds for $f(n)$. For sufficiently large n, $af(n/b) = 3(n/4)\lg(n/4) \le (3/4)n \lg n = cf(n)$ for $c = 3/4$. Consequently, by case 3, the solution to the recurrence is $T(n) = \Theta(n \lg n)$.

Master Theorem : قضیه اصلی

$$T(n) = 2T(n/2) + n \lg n ,$$

even though it has the proper form: $a = 2$, $b = 2$, $f(n) = n \lg n$, and $n^{\log_b a} = n$. It might seem that case 3 should apply, since $f(n) = n \lg n$ is asymptotically larger than $n^{\log_b a} = n$. The problem is that it is not *polynomially* larger. The ratio $f(n)/n^{\log_b a} = (n \lg n)/n = \lg n$ is asymptotically less than n^{ϵ} for any positive constant ϵ. Consequently, the recurrence falls into the gap between case 2 and case 3. (See Exercise 4.4-2 for a solution.)

4.4-2 ★

Show that if $f(n) = \Theta(n^{\log_b a} \lg^k n)$, where $k \geq 0$, then the master recurrence has solution $T(n) = \Theta(n^{\log_b a} \lg^{k+1} n)$. For simplicity, confine your analysis to exact powers of b.

قضیه اصلی : Master Theorem

4.3-1

Use the master method to give tight asymptotic bounds for the following recurrences.

a. $T(n) = 4T(n/2) + n$.

b. $T(n) = 4T(n/2) + n^2$.

c. $T(n) = 4T(n/2) + n^3$.

4.3-2

The recurrence $T(n) = 7T(n/2)+n^2$ describes the running time of an algorithm A. A competing algorithm A' has a running time of $T'(n) = aT'(n/4) + n^2$. What is the largest integer value for a such that A' is asymptotically faster than A?

قضیه اصلی : Master Theorem

4.3-3

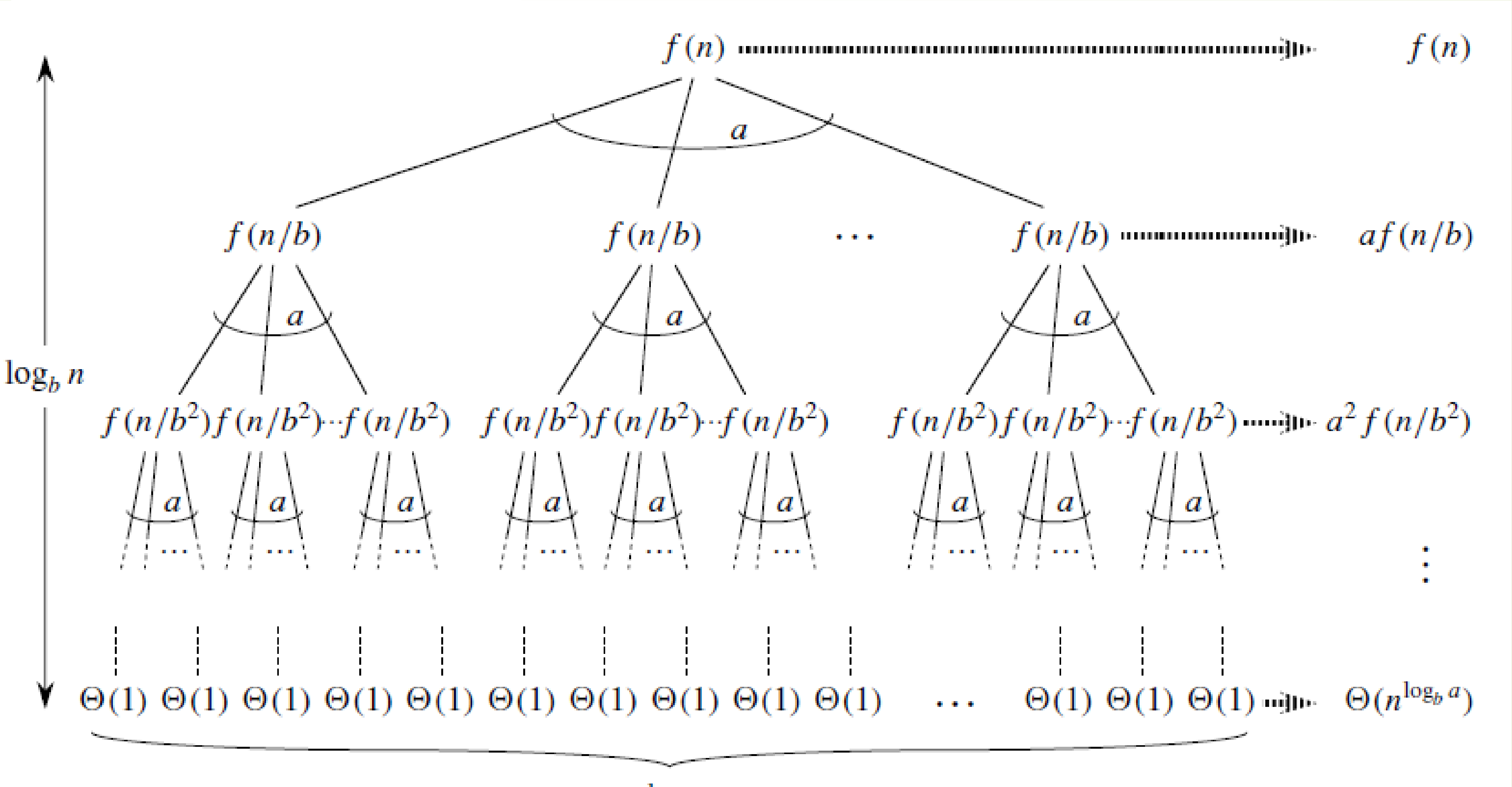
Use the master method to show that the solution to the binary-search recurrence $T(n) = T(n/2) + \Theta(1)$ is $T(n) = \Theta(\lg n)$. (See Exercise 2.3-5 for a description of binary search.)

4.3-4

Can the master method be applied to the recurrence $T(n) = 4T(n/2) + n^2 \lg n$? Why or why not? Give an asymptotic upper bound for this recurrence.

اثبات قضیه اصلی

$$T(n) = aT(n/b) + f(n) ,$$



اثبات قضیه اصلی

$$T(n) = \begin{cases} \Theta(1) & \text{if } n = 1\,, \\ aT(n/b) + f(n) & \text{if } n = b^i\,, \end{cases}$$

where i is a positive integer. Then

$$T(n) = \Theta(n^{\log_b a}) + \sum_{j=0}^{\log_b n - 1} a^j f(n/b^j)\,.$$

اثبات قضیه اصلی

Lemma 4.3

Let $a \ge 1$ and $b > 1$ be constants, and let $f(n)$ be a nonnegative function defined on exact powers of b. A function $g(n)$ defined over exact powers of b by

$$g(n) = \sum_{j=0}^{\log_b n - 1} a^j f(n/b^j) \qquad (4.7)$$

can then be bounded asymptotically for exact powers of b as follows.

1. If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $g(n) = O(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a})$, then $g(n) = \Theta(n^{\log_b a} \lg n)$.
3. If $af(n/b) \le cf(n)$ for some constant $c < 1$ and for all $n \ge b$, then $g(n) = \Theta(f(n))$.

اثبات قضیه اصلی

If $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then $g(n) = O(n^{\log_b a})$.

Proof For case 1, we have $f(n) = O(n^{\log_b a - \epsilon})$, which implies that $f(n/b^j) = O((n/b^j)^{\log_b a - \epsilon})$. Substituting into equation (4.7) yields

$$g(n) = O\left(\sum_{j=0}^{\log_b n - 1} a^j \left(\frac{n}{b^j}\right)^{\log_b a - \epsilon}\right) . \tag{4.8}$$

اثبات قضیه اصلی

$$
\begin{aligned}
\sum_{j=0}^{\log_b n-1} a^j \left(\frac{n}{b^j}\right)^{\log_b a-\epsilon} &= n^{\log_b a-\epsilon} \sum_{j=0}^{\log_b n-1} \left(\frac{ab^\epsilon}{b^{\log_b a}}\right)^j \\
&= n^{\log_b a-\epsilon} \sum_{j=0}^{\log_b n-1} (b^\epsilon)^j \\
&= n^{\log_b a-\epsilon} \left(\frac{b^{\epsilon \log_b n}-1}{b^\epsilon-1}\right) \\
&= n^{\log_b a-\epsilon} \left(\frac{n^\epsilon-1}{b^\epsilon-1}\right) .
\end{aligned}
$$

اثبات قضیه اصلی

- اثبات حالت (۲)

$$f(n) = \theta\left(n^{\log_b^a}\right) \rightarrow T(n) = \theta(n^{\log_b^a} \log_b^n)$$

$$f(n) = \theta\left(n^{\log_b^a}\right) \rightarrow \ \exists \ \ C_1, C_2, n_0 > 0 \quad : \quad \forall n \geq n_0 \ \ C_1 n^{\log_b^a} \leq f(n) \leq C_2 n^{\log_b^a}$$

$$C_1 \sum_{j=0}^{\log_b^n - 1} a^j \left({}^{n}/_{b^j}\right)^{\log_b^a} \leq \sum_{j=0}^{\log_b^n - 1} \leq a^j f\left({}^{n}/_{b^j}\right) \leq C_2 \sum_{j=0}^{\log_b^n - 1} a^j \left({}^{n}/_{b^j}\right)^{\log_b^a}$$

$$\sum a^j f\left({}^{n}/_{b^j}\right) = \theta\left(\sum_{j=0}^{\log_b^n - 1} a^j \left({}^{n}/_{b^j}\right)^{\log_b^a}\right) \Rightarrow \theta\left(n^{\log_b^a} \sum_{j=0}^{\log_b^n - 1} \frac{a^j}{(b^j)^{\log_b^a}}\right)$$

$$T(n) = \theta(n^{\log_b^a} \log n)$$

اثبات قضیه اصلی

- اثبات حالت (۳)

$$\exists \quad \varepsilon > 0 \qquad f(n) = \Omega\left(n^{\log_b^a + \varepsilon}\right) \rightarrow \quad \exists \quad C > 1: a\, f(n/b) \leq C\, f(n)$$

$$\rightarrow T\,(n) = \theta(f(n))$$

$$* \quad \sum_{j=0}^{\log_b^n - 1} a^j f\left(\frac{n}{b^j}\right) = a^0 f(n) + a^1 f(n/b) + \cdots = \Omega(f(n))$$

$$a\, f(n/b) \leq c\, f(n) \rightarrow f(n) \geq \frac{a}{c} f\,(n/b)$$

$$\geq \frac{a}{c}\left(\frac{a}{c} f\left({}^{n}/{}_{b^2}\right)\right) = (\frac{a}{b})^2 f(\,{}^{n}/{}_{b^2})$$

$$\vdots$$

$$f(n) \geq (\frac{a}{c})^j f\left({}^{n}/{}_{b^j}\right) \xrightarrow{a^j} f\left({}^{n}/{}_{b^j}\right) \leq (\frac{c}{a})^j f(n)$$

اثبات قضیه اصلی

$$\sum_{j=0}^{\log_b^n - 1} a^j f\left({}^{n}/_{b^j}\right) \leq f(n) \sum_{j=0}^{\log_b^n - 1} c^j \leq f(n) \sum_{j=0}^{\infty} C^j = f(n) \frac{1}{1 - C}$$

$$** \rightarrow \sum_{j=0}^{\log_b^n - 1} a^j f\left({}^{n}/_{b^j}\right) \leq C' f(n) \Rightarrow \sum a^j f\left({}^{n}/_{b^j}\right) = O(f(n))$$

از $*.** \longrightarrow \sum a^j f\left({}^{n}/_{b^j}\right) = \theta\left(f(n)\right)$

قضیه اصلی

$T(n) = 3T(n/2) + n \lg n.$

$T(n) = 5T(n/5) + n/\lg n.$

$T(n) = 4T(n/2) + n^2\sqrt{n}.$

$T(n) = 3T(n/3 + 5) + n/2.$

$T(n) = 2T(n/2) + n/\lg n.$

$T(n) = T(n/2) + T(n/4) + T(n/8) + n.$

$T(n) = T(n-1) + 1/n.$

$T(n) = T(n-1) + \lg n.$

$T(n) = T(n-2) + 2\lg n.$

$T(n) = \sqrt{n}T(\sqrt{n}) + n.$

$T(n) = 2T(n/2) + n^3.$

$T(n) = T(9n/10) + n.$

$T(n) = 16T(n/4) + n^2.$

$T(n) = 7T(n/3) + n^2.$

$T(n) = 7T(n/2) + n^2.$

$T(n) = 2T(n/4) + \sqrt{n}.$

$T(n) = T(n-1) + n.$

$T(n) = T(\sqrt{n}) + 1.$