

Analysis of Divide and Conquer Algorithms

Marc Moreno Maza

University of Western Ontario, London, Ontario (Canada)

CS3101

Plan

- 1 Review of Complexity Notions
- 2 Divide-and-Conquer Recurrences

Plan

- 1 Review of Complexity Notions
- 2 Divide-and-Conquer Recurrences

Orders of magnitude

Let f , g et h be functions from \mathbb{N} to \mathbb{R} .

- We say that $g(n)$ is in the **order of magnitude** of $f(n)$ and we write $f(n) \in \Theta(g(n))$ if there exist two strictly positive constants c_1 and c_2 such that for n big enough we have

$$0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n). \quad (1)$$

- We say that $g(n)$ is an **asymptotic upper bound** of $f(n)$ and we write $f(n) \in \mathcal{O}(g(n))$ if there exists a strictly positive constants c_2 such that for n big enough we have

$$0 \leq f(n) \leq c_2 g(n). \quad (2)$$

- We say that $g(n)$ is an **asymptotic lower bound** of $f(n)$ and we write $f(n) \in \Omega(g(n))$ if there exists a strictly positive constants c_1 such that for n big enough we have

$$0 \leq c_1 g(n) \leq f(n). \quad (3)$$

Examples

- With $f(n) = \frac{1}{2}n^2 - 3n$ and $g(n) = n^2$ we have $f(n) \in \Theta(g(n))$. Indeed we have

$$c_1 n^2 \leq \frac{1}{2}n^2 - 3n \leq c_2 n^2. \quad (4)$$

for $n \geq 12$ with $c_1 = \frac{1}{4}$ and $c_2 = \frac{1}{2}$.

- Assume that there exists a positive integer n_0 such that $f(n) > 0$ and $g(n) > 0$ for every $n \geq n_0$. Then we have

$$\max(f(n), g(n)) \in \Theta(f(n) + g(n)). \quad (5)$$

Indeed we have

$$\frac{1}{2}(f(n) + g(n)) \leq \max(f(n), g(n)) \leq (f(n) + g(n)). \quad (6)$$

- Assume a and b are positive real constants. Then we have

$$(n + a)^b \in \Theta(n^b). \quad (7)$$

Indeed for $n \geq a$ we have

$$0 \leq n^b \leq (n + a)^b \leq (2n)^b. \quad (8)$$

Hence we can choose $c_1 = 1$ and $c_2 = 2^b$.



Properties

- $f(n) \in \Theta(g(n))$ holds iff $f(n) \in \mathcal{O}(g(n))$ and $f(n) \in \Omega(g(n))$ hold together.
- Each of the predicates $f(n) \in \Theta(g(n))$, $f(n) \in \mathcal{O}(g(n))$ and $f(n) \in \Omega(g(n))$ define a reflexive and transitive binary relation among the \mathbb{N} -to- \mathbb{R} functions. Moreover $f(n) \in \Theta(g(n))$ is symmetric.
- We have the following **transposition formula**

$$f(n) \in \mathcal{O}(g(n)) \iff g(n) \in \Omega(f(n)). \quad (9)$$

In practice \in is replaced by $=$ in each of the expressions $f(n) \in \Theta(g(n))$, $f(n) \in \mathcal{O}(g(n))$ and $f(n) \in \Omega(g(n))$. Hence, the following

$$f(n) = h(n) + \Theta(g(n)) \quad (10)$$

means

$$f(n) - h(n) \in \Theta(g(n)). \quad (11)$$



Another example

Let us give another fundamental example. Let $p(n)$ be a (univariate) polynomial with degree $d > 0$. Let a_d be its leading coefficient and assume $a_d > 0$. Let k be an integer. Then we have

- if $k \geq d$ then $p(n) \in \mathcal{O}(n^k)$,
- if $k \leq d$ then $p(n) \in \Omega(n^k)$,
- if $k = d$ then $p(n) \in \Theta(n^k)$.

Exercise: Prove the following

$$\sum_{k=1}^{k=n} k \in \Theta(n^2). \quad (12)$$



Plan

- Review of Complexity Notions
- Divide-and-Conquer Recurrences



Divide-and-Conquer Algorithms

Divide-and-conquer algorithms proceed as follows.

Divide the input problem into sub-problems.

Conquer on the sub-problems by solving them directly if they are small enough or proceed recursively.

Combine the solutions of the sub-problems to obtain the solution of the input problem.

Equation satisfied by $T(n)$. Assume that the size of the input problem increases with an integer n . Let $T(n)$ be the time complexity of a divide-and-conquer algorithm to solve this problem. Then $T(n)$ satisfies an equation of the form:

$$T(n) = a T(n/b) + f(n). \tag{13}$$

where $f(n)$ is the cost of the combine-part, $a \geq 1$ is the number of recursively calls and n/b with $b > 1$ is the size of a sub-problem.



Tree associated with a divide-and-conquer recurrence

Labeled tree associated with the equation. Assume n is a power of b , say $n = b^p$. To solve the equation

$$T(n) = a T(n/b) + f(n).$$

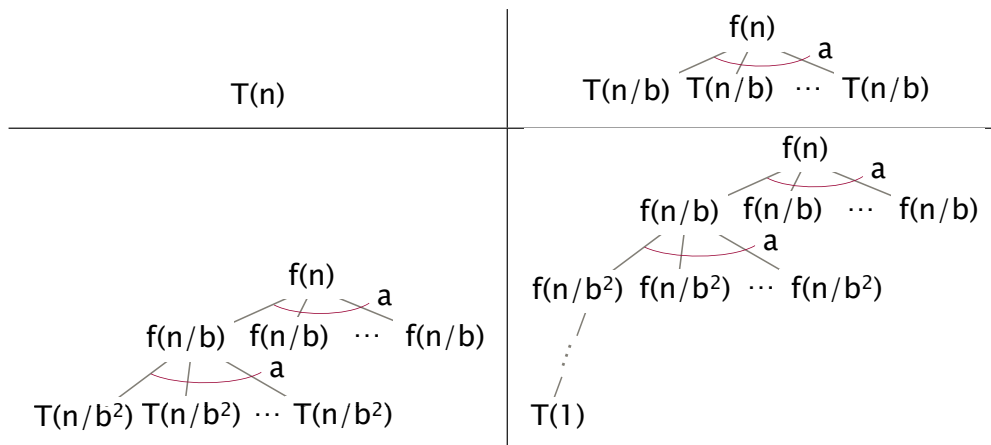
we can associate a labeled tree $\mathcal{A}(n)$ to it as follows.

- (1) If $n = 1$, then $\mathcal{A}(n)$ is reduced to a single leaf labeled $T(1)$.
- (2) If $n > 1$, then the root of $\mathcal{A}(n)$ is labeled by $f(n)$ and $\mathcal{A}(n)$ possesses a labeled sub-trees all equal to $\mathcal{A}(n/b)$.

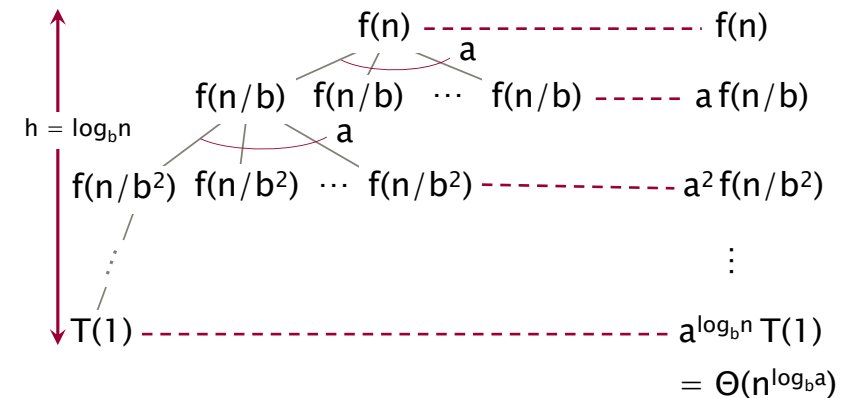
The labeled tree $\mathcal{A}(n)$ associated with $T(n) = a T(n/b) + f(n)$ has height $p + 1$. Moreover the sum of its labels is $T(n)$.



Solving divide-and-conquer recurrences (1/2)

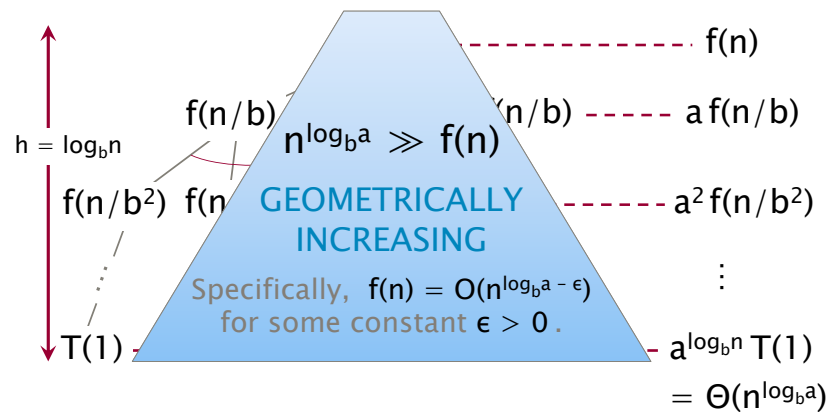


Solving divide-and-conquer recurrences (2/2)

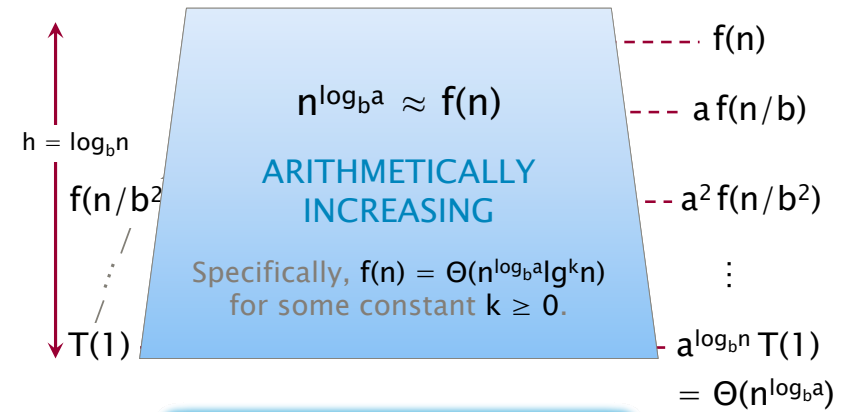


IDEA: Compare $n^{\log_b a}$ with $f(n)$.

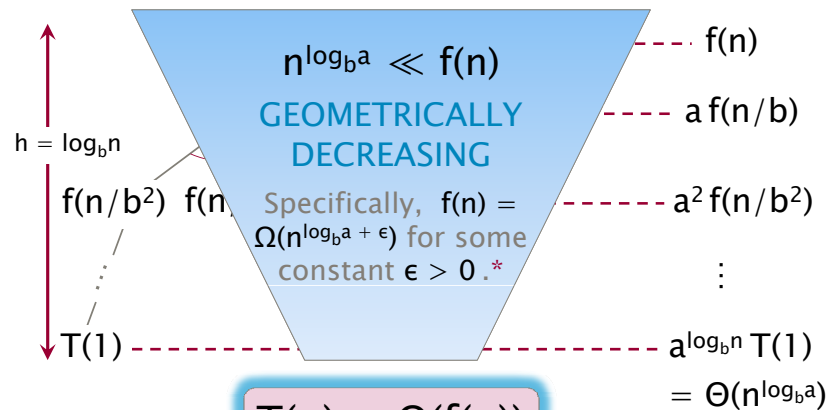


Master Theorem: case $n^{\log_b a} \gg f(n)$ 

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

Master Theorem: case $f(n) \in \Theta(n^{\log_b a} \lg^k n)$ 

$$T(n) = \Theta(n^{\log_b a} \lg^{k+1} n)$$

Master Theorem: case where $f(n) \gg n^{\log_b a}$ 

$$T(n) = \Theta(f(n))$$

*and $f(n)$ satisfies the **regularity condition** that $a f(n/b) \leq c f(n)$ for some constant $c < 1$.

More examples

- Consider the relation:

$$T(n) = 2 T(n/2) + n^2. \quad (14)$$

We obtain:

$$T(n) = n^2 + \frac{n^2}{2} + \frac{n^2}{4} + \frac{n^2}{8} + \dots + \frac{n^2}{2^p} + n T(1). \quad (15)$$

Hence we have:

$$T(n) \in \Theta(n^2). \quad (16)$$

- Consider the relation:

$$T(n) = 3T(n/3) + n. \quad (17)$$

We obtain:

$$T(n) \in \Theta(\log_3(n)n). \quad (18)$$

Master Theorem when $b = 2$

Let $a > 0$ be an integer and let $f, T : \mathbb{N} \rightarrow \mathbb{R}_+$ be functions such that

- (i) $f(2n) \geq 2f(n)$ and $f(n) \geq n$.
- (ii) If $n = 2^p$ then $T(n) \leq aT(n/2) + f(n)$.

Then for $n = 2^p$ we have

- (1) if $a = 1$ then

$$T(n) \leq (2 - 2/n)f(n) + T(1) \in \mathcal{O}(f(n)), \quad (19)$$

- (2) if $a = 2$ then

$$T(n) \leq f(n) \log_2(n) + T(1)n \in \mathcal{O}(\log_2(n)f(n)), \quad (20)$$

- (3) if $a \geq 3$ then

$$T(n) \leq \frac{2}{a-2} \left(n^{\log_2(a)-1} - 1 \right) f(n) + T(1)n^{\log_2(a)} \in \mathcal{O}(f(n)n^{\log_2(a)-1}). \quad (21)$$

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↺

Master Theorem when $b = 2$

Indeed

$$\begin{aligned} T(2^p) &\leq aT(2^{p-1}) + f(2^p) \\ &\leq a[aT(2^{p-2}) + f(2^{p-1})] + f(2^p) \\ &= a^2T(2^{p-2}) + af(2^{p-1}) + f(2^p) \\ &\leq a^2[aT(2^{p-3}) + f(2^{p-2})] + af(2^{p-1}) + f(2^p) \\ &= a^3T(2^{p-3}) + a^2f(2^{p-2}) + af(2^{p-1}) + f(2^p) \\ &\leq a^pT(1) + \sum_{j=0}^{p-1} a^j f(2^{p-j}) \end{aligned} \quad (22)$$

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↺

Master Theorem when $b = 2$

Moreover

$$\begin{aligned} f(2^p) &\geq 2f(2^{p-1}) \\ f(2^p) &\geq 2^2f(2^{p-2}) \\ &\vdots \\ f(2^p) &\geq 2^j f(2^{p-j}) \end{aligned} \quad (23)$$

Thus

$$\sum_{j=0}^{p-1} a^j f(2^{p-j}) \leq f(2^p) \sum_{j=0}^{p-1} \left(\frac{a}{2}\right)^j. \quad (24)$$

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↺

Master Theorem when $b = 2$

Hence

$$T(2^p) \leq a^p T(1) + f(2^p) \sum_{j=0}^{p-1} \left(\frac{a}{2}\right)^j. \quad (25)$$

For $a = 1$ we obtain

$$\begin{aligned} T(2^p) &\leq T(1) + f(2^p) \sum_{j=0}^{p-1} \left(\frac{1}{2}\right)^j \\ &= T(1) + f(2^p) \frac{1 - \frac{1}{2^p}}{\frac{1}{2} - 1} \\ &= T(1) + f(n)(2 - 2/n). \end{aligned} \quad (26)$$

For $a = 2$ we obtain

$$\begin{aligned} T(2^p) &\leq 2^p T(1) + f(2^p)p \\ &= nT(1) + f(n) \log_2(n). \end{aligned} \quad (27)$$

◀ ▶ ⏪ ⏩ ⏴ ⏵ ⏶ ⏷ ⏸ ⏹ ⏺ ⏻ ⏼ ⏽ ⏾ ⏿ 🔍 ↺

Master Theorem cheat sheet

For $a \geq 1$ and $b > 1$, consider again the equation

$$T(n) = a T(n/b) + f(n). \quad (28)$$

- We have:

$$(\exists \varepsilon > 0) f(n) \in O(n^{\log_b a - \varepsilon}) \implies T(n) \in \Theta(n^{\log_b a}) \quad (29)$$

- We have:

$$(\exists \varepsilon > 0) f(n) \in \Theta(n^{\log_b a} \log^k n) \implies T(n) \in \Theta(n^{\log_b a} \log^{k+1} n) \quad (30)$$

- We have:

$$(\exists \varepsilon > 0) f(n) \in \Omega(n^{\log_b a + \varepsilon}) \implies T(n) \in \Theta(f(n)) \quad (31)$$

Acknowledgements

- Charles E. Leiserson (MIT) for providing me with the sources of its lecture notes.

Master Theorem quizz!

- $T(n) = 4T(n/2) + n$
- $T(n) = 4T(n/2) + n^2$
- $T(n) = 4T(n/2) + n^3$
- $T(n) = 4T(n/2) + n^2/\log n$