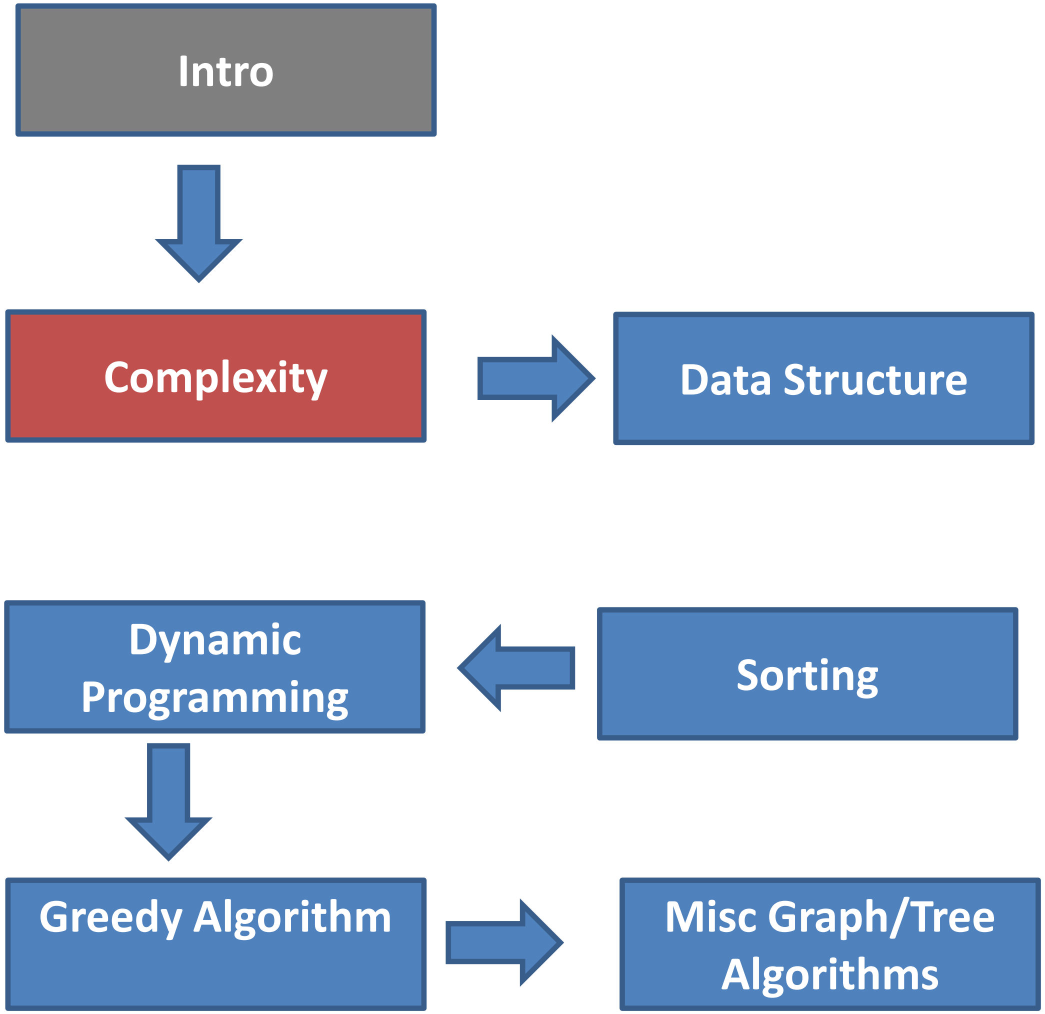
An Introduction to Algorithms

By

Hossein Rahmani

h_rahmani@iust.ac.ir

http://webpages.iust.ac.ir/h_rahmani/


Intro
Complexity
Data Structure
Trees
Hash Functions
Sorting
Dynamic Programming
Greedy Algorithm
Misc Graph/Tree Algorithms

Substitution method

- *The most general method:*
  1. ***Guess*** the form of the solution.
  2. ***Verify*** (or refine) by induction.
  3. ***Solve*** for constants.
- ***Example:*** $T(n) = 4T(n/2) + 100n$
  - [Assume that $T(1) = \Theta(1)$.]
  - Guess $O(n^3)$. (Prove O and Ω separately.)
  - Assume that $T(k) \leq ck^3$ for $k < n$.
  - Prove $T(n) \leq cn^3$ by induction.

Example of substitution

$$
\begin{aligned}
T(n) &= 4T(n/2) + 100n \\
&\le 4c(n/2)^3 + 100n \\
&= (c/2)n^3 + 100n \\
&= cn^3 - ((c/2)n^3 - 100n) \quad \longleftarrow \textit{desired} - \textit{residual} \\
&\le cn^3 \quad \longleftarrow \textit{desired}
\end{aligned}
$$

whenever $(c/2)n^3 - 100n \ge 0$, for example, if $c \ge 200$ and $n \ge 1$.

residual

A tighter upper bound?

We shall prove that $T(n) = O(n^2)$.

Assume that $T(k) \leq ck^2$ for $k < n$:

$$T(n) = 4T(n/2) + 100n$$

$$\leq cn^2 + 100n$$

$$\leq cn^2$$

for ***no*** choice of $c > 0$. Lose!

Substitution Method

- $T(n) = 2\,T(\lfloor n/2 \rfloor) + n$

- Guess: $O(n \lg n)$
- Verify
  - Inductive Hypothesis: $T(n) \le c\, n \lg n$ for appropriate choice of $c > 0$
  - Prove that $T(n) \le c\, n \lg n$ for appropriate choice of $c > 0$

Use induction:

Assume $T(\lfloor n/2 \rfloor) \le c \lfloor n/2 \rfloor \lg(\lfloor n/2 \rfloor)$ holds

Show $\quad T(n) \le c\, n \lg n$

Substitution Method

$T(n) = 2\,T(\lfloor n/2 \rfloor) + n$

$$
\begin{aligned}
T(n) &\le 2\,c\lfloor n/2 \rfloor \lg(\lfloor n/2 \rfloor) + n && \text{apply IH} \\
&\le c\,n\,\lg(n/2) + n \\
&= c\,n\,\lg n - c\,n\,\lg 2 + n \\
&= c\,n\,\lg n - c\,n + n \\
&\le c\,n\,\lg n \quad \text{when } c \ge 1
\end{aligned}
$$

Substitution Method – Changing Variables

- $T(n) = 2\,T(\lfloor \sqrt{n} \rfloor) + \lg n$ ➔ a difficult recurrence

- Rename m as lgn yields
  $T(2^m) = 2\,T(2^{m/2}) + m$
- Rename $S(m) = T(2^m)$
  $S(m) = 2\,S(m/2) + m$
- Similar to our previous recurrence
  ➔ $O(m \lg m)$
- Change back S(m) to T(n)
  $T(n) = T(2^m) = S(m) = O(m \lg m)$
  ➔ **O(lgn lg lg n)**

Recursion-tree method

- A recursion tree models the costs (time) of a recursive execution of an algorithm.
- The recursion tree method is good for generating guesses for the substitution method.
- The recursion-tree method promotes intuition, however.

Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:

Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:

$$T(n)$$

Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:

$$n^2$$

$$T(n/4) \qquad\qquad T(n/2)$$

Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:

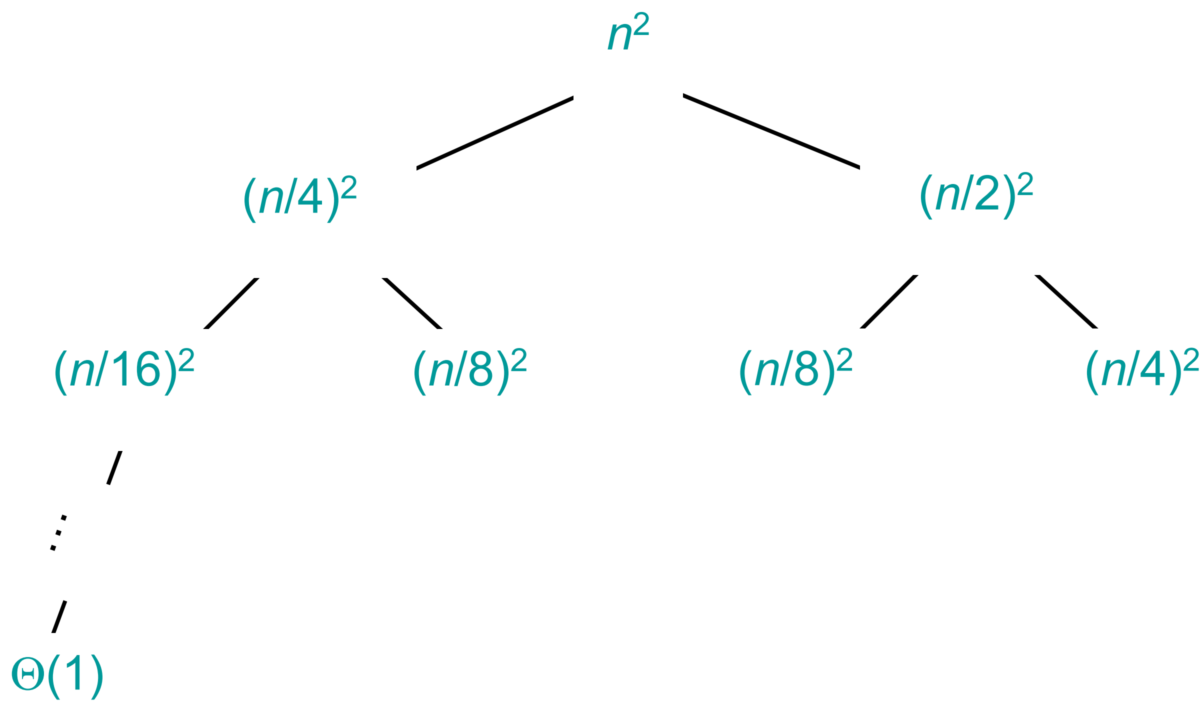
n^2

$(n/4)^2$ $\quad\quad\quad\quad$ $(n/2)^2$

$T(n/16)$ $\quad$ $T(n/8)$ $\quad$ $T(n/8)$ $\quad$ $T(n/4)$

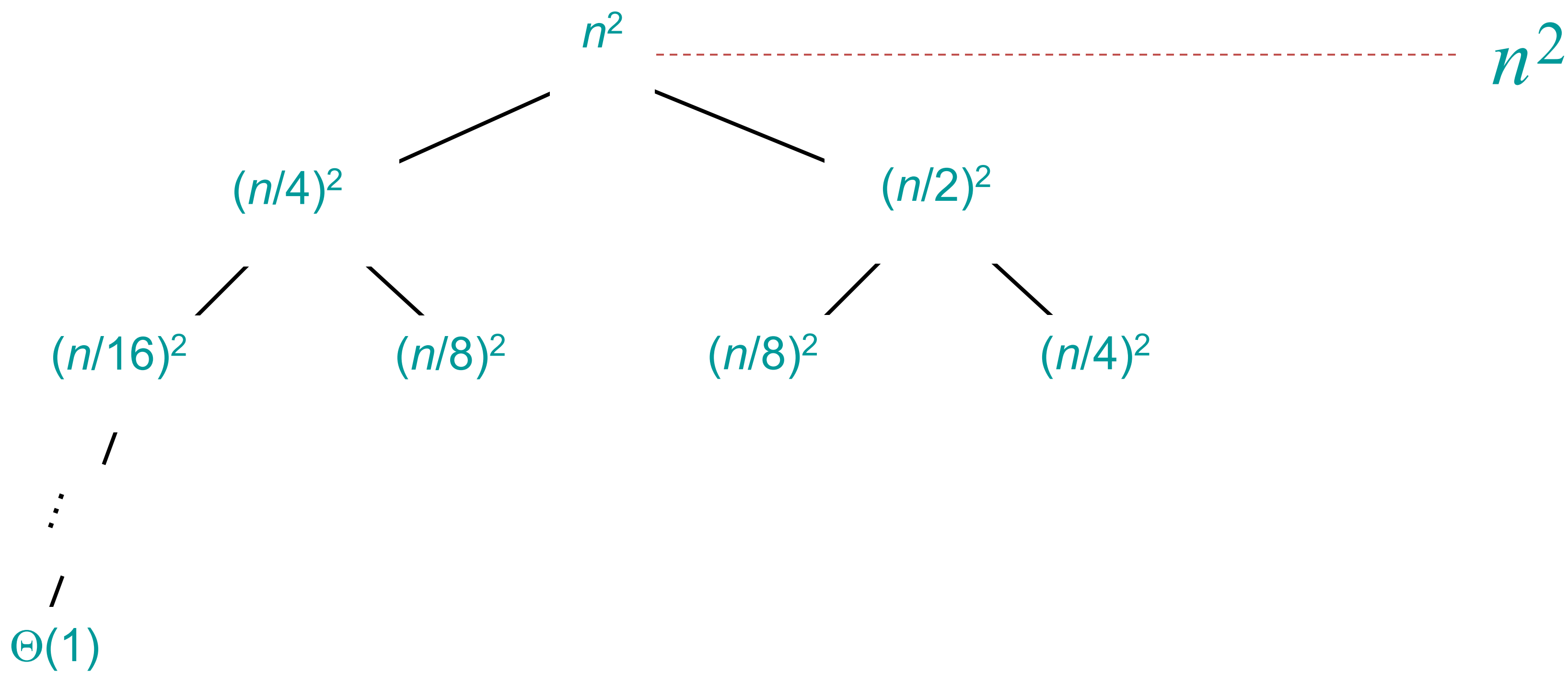
Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:



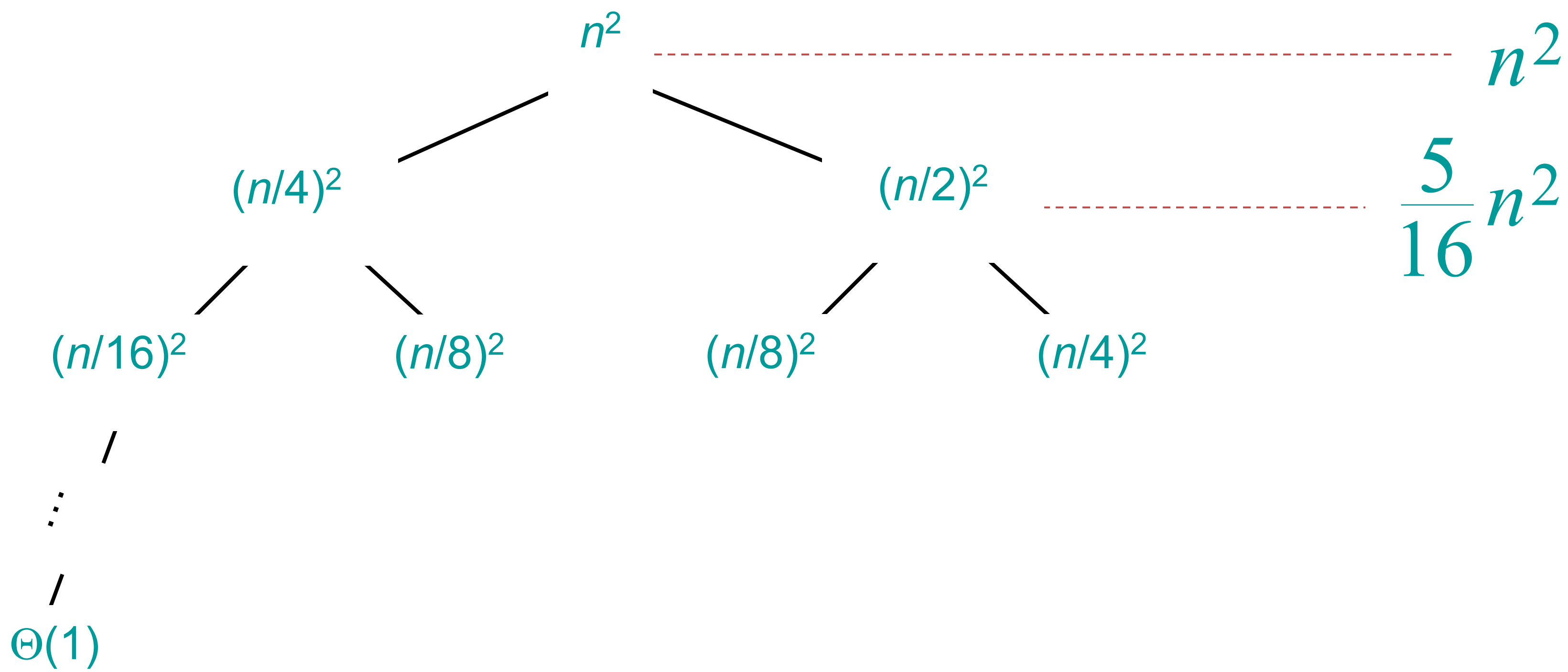
Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:



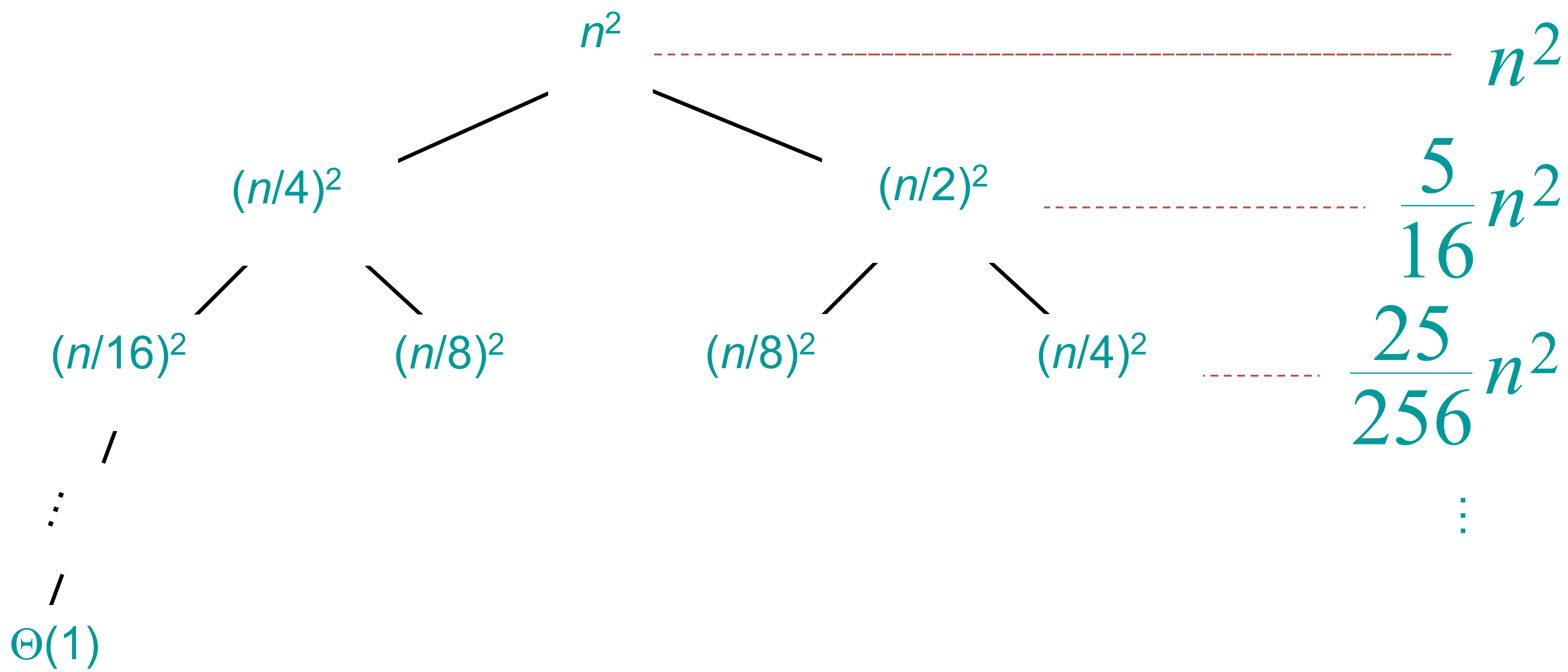
Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:



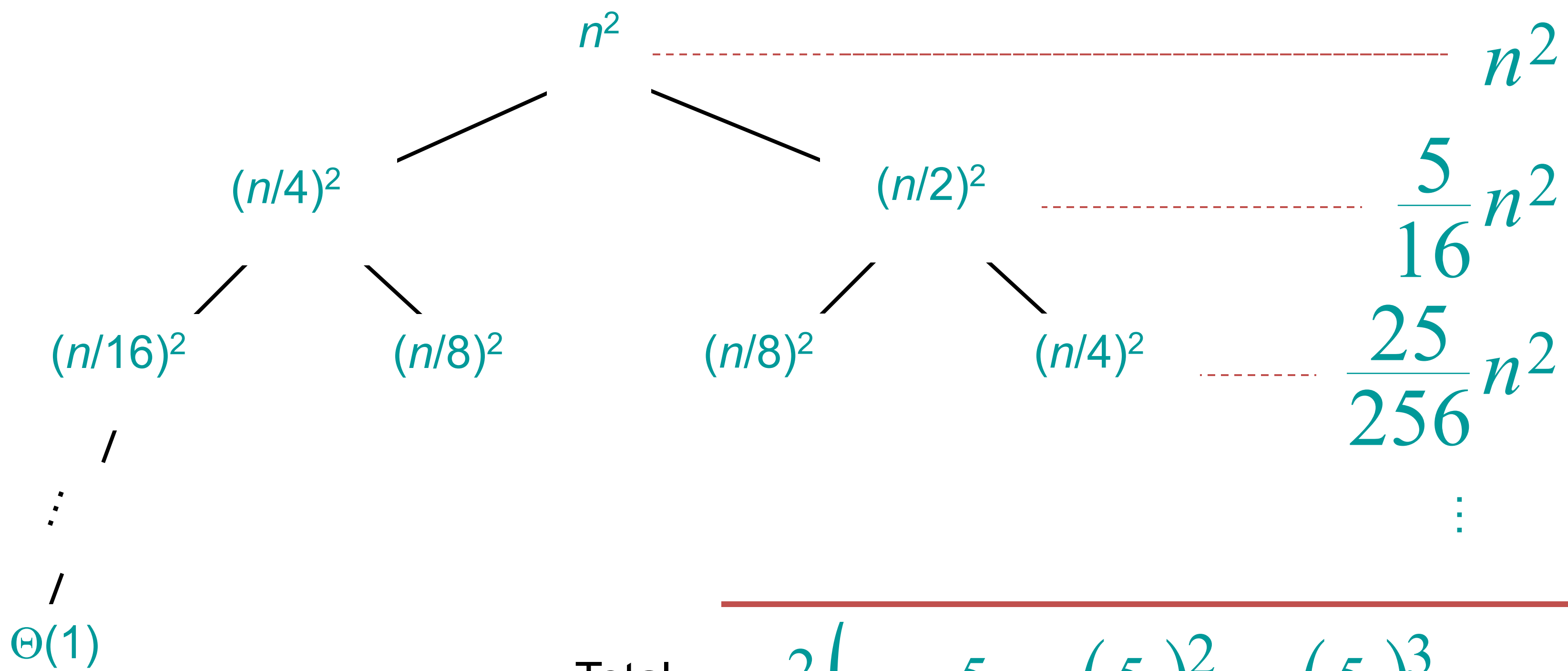
Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:



Example of recursion tree

Solve $T(n) = T(n/4) + T(n/2) + n^2$:



$$\text{Total} = n^2\left(1 + \frac{5}{16} + \left(\frac{5}{16}\right)^2 + \left(\frac{5}{16}\right)^3 + \cdots\right)$$

$= \Theta(n^2)$

geometric series

Appendix: geometric series

$$1+x+x^2+\cdots+x^n=\frac{1-x^{n+1}}{1-x}$$ for $x \neq 1$

$$1+x+x^2+\cdots=\frac{1}{1-x}$$ for $|x| < 1$

The master method

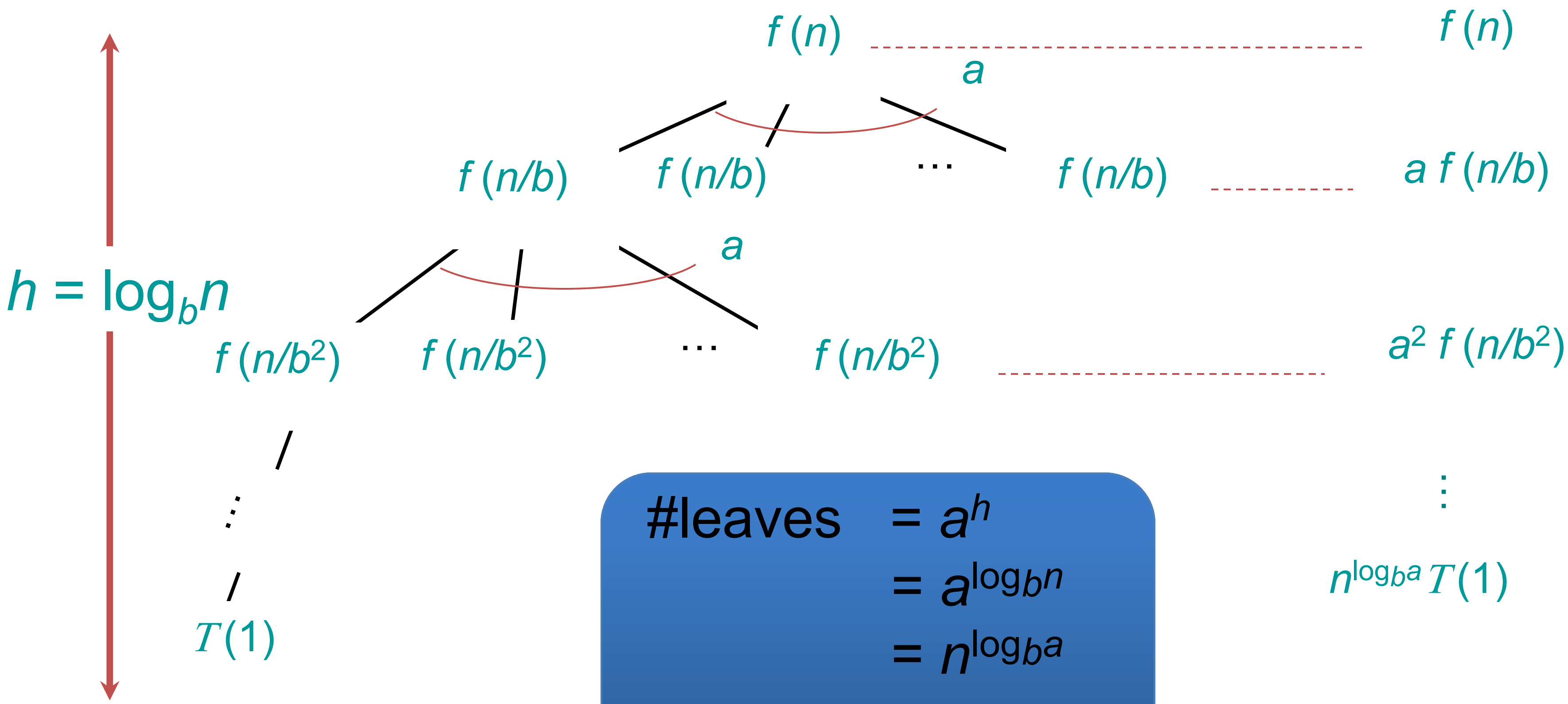
The master method applies to recurrences of the form

$$T(n) = a\,T(n/b) + f(n)\ ,$$

where $a \geq 1$, $b > 1$, and f is asymptotically positive.

Idea of master theorem

Recursion tree:



$$\begin{aligned} \#\text{leaves} &= a^h \\ &= a^{\log_b n} \\ &= n^{\log_b a} \end{aligned}$$

Three common cases

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

1. If $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$.

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

Three common cases

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

2. If $f(n) = \Theta(n^{\log_b a})$

   Solution: $T(n) = \Theta(n^{\log_b a} \lg n)$.

Three common cases (cont.)

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

3. $f(n) = \Omega(n^{\log_b a + \varepsilon})$ for some constant $\varepsilon > 0$.

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

   Solution: $T(n) = \Theta(\,f(n)\,)$.

Master Theorem - Binary Search

$$T(n) = 1\,T(n/2) + \Theta(1)$$

subproblems

subproblem size

work dividing and combining

$$n^{\log_b^a} = n^{\log_2^1} = n^0 = 1 \quad =>$$

CASE 2 (k=0) ➔ **T(n) = Θ(lgn)**

Powering a Number

- **Problem:** Compute a^n, where n is in $\mathbb{N}$.
- **Naive algorithm:** $\Theta(n)$
- **Divide-and-conquer algorithm:**

$$a^n = \begin{cases} a^{n/2} \cdot a^{n/2} & \text{if } n \text{ is even} \\ a^{(n-1)/2} \cdot a^{(n-1)/2} \cdot a & \text{if } n \text{ is odd} \end{cases}$$

$$T(n) = T(n/2) + \Theta(1) \quad \Rightarrow \quad T(n) = \Theta(\lg n).$$

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

$a = 4$, $b = 2 \Rightarrow n^{\log_b a} = n^2$; $f(n) = n$.

$f(n) = O(n^{2-\varepsilon})$ for $\varepsilon = 1$ => Case 1

$\therefore\ T(n) = \Theta(n^2)$.

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

$a = 4$, $b = 2 \Rightarrow n^{\log_b a} = n^2$; $f(n) = n^2$.

$f(n) = \Theta(n^2)$ => Case 2

$\therefore\ T(n) = \Theta(n^2 \lg n)$.

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

$a = 4$, $b = 2 \Rightarrow n^{\log_b a} = n^2$; $f(n) = n^3$.

$f(n) = \Omega(n^{2+\varepsilon})$ for $\varepsilon = 1$ => Case 3

and $4(cn/2)^3 \leq cn^3$ (reg. cond.) for $c = 1/2$.

$\therefore\ T(n) = \Theta(n^3)$.

Quiz

- Make the group of 2
- For each of the following recurrences, give an expression for the runtime T (n) if the recurrence can be solved with the Master Theorem. Otherwise, indicate that the Master Theorem does not apply.

1. $T(n) = 3T(n/2) + n^2$
2. $T(n) = 4T(n/2) + n^2$
3. $T(n) = T(n/2) + 2^n$
4. $T(n) = 2^n T(n/2) + n^n$
5. $T(n) = 16T(n/4) + n$
6. $T(n) = 2T(n/2) + n \log n$
7. $T(n) = 2T(n/2) + n/\log n$
8. $T(n) = 2T(n/4) + n^{0.51}$
9. $T(n) = 0.5T(n/2) + 1/n$
10. $T(n) = 16T(n/4) + n!$
11. $T(n) = \sqrt{2}T(n/2) + \log n$
12. $T(n) = 3T(n/2) + n$