Solving recurrences

Gou Guanglei(苟光磊)
ggl@cqut.edu.cn

- Divide and conquer design paradigm
  - Divide: divide a problem into subproblems
  - Conquer: solve the problems recursively
  - Combine: combine the subproblems solutions appropriately

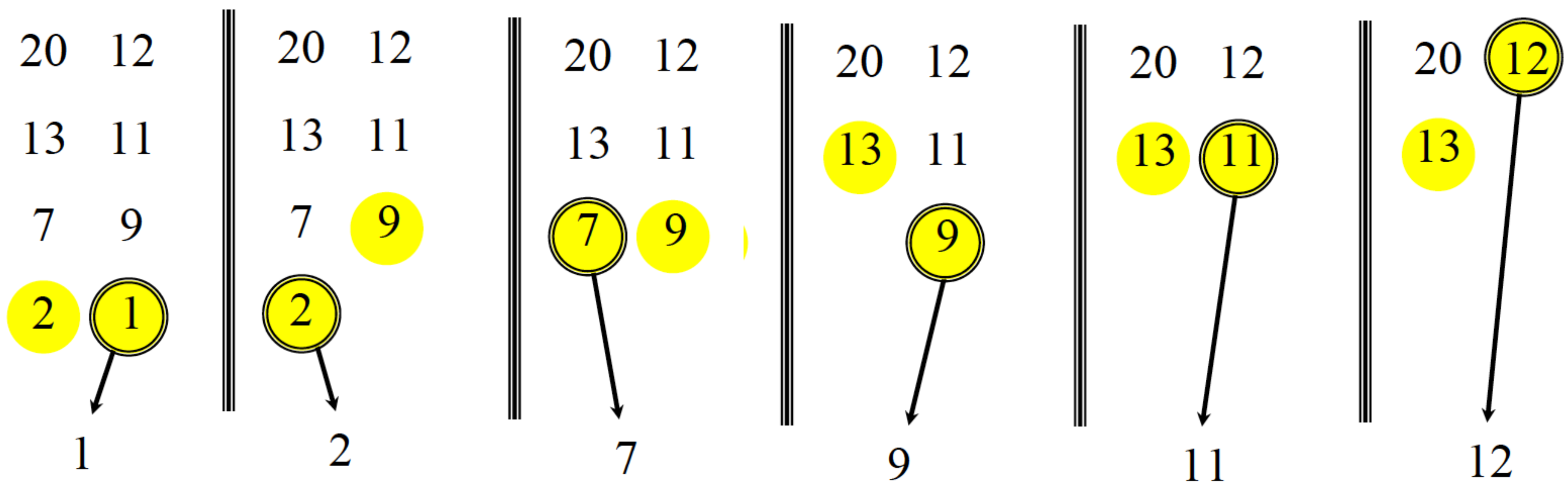
Merge sort

MERGEMERGE-SORT$A[1 \,..\, n]$

1. If $n = 1$, done.
2. Recursively sort $A[\,1 \,..\, \lceil n/2 \rceil]$ and $A[\,\lceil n/2 \rceil + 1 \,..\, n\,]$.
3. "***Merge***" the 2sorted lists.

Key subroutine: **MERGE**

Merging two sorted arrays



Time = $\Theta(n)$ to merge a total of n elements (linear time).

http://algs4.cs.princeton.edu

2.2 MERGING DEMO

Analysis

$\boldsymbol{T(n)}$	**MERGEMERGE-SORT** $A[1\ ..\ n]$
$\boldsymbol{\Theta(1)}$	1. If $n = 1$, done.
$\boldsymbol{2T(n/2)}$	2. Recursively sort $A[\,1\ ..\ \lceil n/2 \rceil]$ and $A[\lceil n/2 \rceil + 1\ ..\ n\,]$.
$\boldsymbol{\Theta(n)}$	3. "***Merge***" the 2 sorted lists.

$$T(n) = \Theta(1) \quad \text{if } n = 1;$$
$$T(n) = 2T(n/2) + \Theta(n) \quad \text{if } n > 1.$$

Analysis of merge sort

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

 # subproblems

subproblem size

work dividing and combining

solving recurrences

- Solve by unrolling
- Substitution method
- Recursion tree
- Master theorem

solve by unrolling

- For example, selection sort recursively:
    - $T(n) = cn + T(n-1)$
    - $= cn + c(n-1) + T(n-2)$
    - $= \ldots$
    - $= cn + c(n-1) + c(n-2) + \ldots + c$ $\quad T(n) = \Theta(n^2)$
  - for each term, at most cn
  - for the top $n/2$ terms, at least $cn/2$
  - **$(n/2)(cn/2) \le T(n) \le cn^2$**

Exercise 1. A method for solving recurrence relation is to expand the recurrence a few times, until a pattern emerges. For instance, let's start with the familiar $T(n) = 2T(n/2) + O(n)$. Think of $O(n)$ as being $\leq cn$ for some constant c, so: $T(n) \leq 2T(n/2) + cn$. By repeatedly applying this rule, we can bound $T(n)$ in terms of $T(n/2)$, $T(n/4)$, and $T(n/8)$, and so on, at each step getting closer to the value of $T(1)$, or $O(1)$.

$$
\begin{aligned}
T(n) &\leq 2T(n/2) + cn \\
&\leq 2[2T(n/4) + cn/2] + cn = 4T(n/4) + 2cn \\
&\leq 4[2T(n/8) + cn/4] + 2cn = 8T(n/8) + 3cn \\
&\leq 8[2T(n/16) + cn/8] + 3cn = 16T(n/16) + 4cn \\
&\vdots
\end{aligned}
$$

A pattern is emerging... the general term is

$$T(n) \leq 2^k T(n/2^k) + kcn$$

Plugging in $k = \log_2 n$, we get $T(n) \leq nT(1) + cn\log_2 n = O(n\log n)$.

Substitution method

- guess the solution
- use mathematical induction to find the constant and show that the solution works
- show the lower and the upper bound separately

Recursion tree

- can be used to provide a good guess for substitution method
- each node represents the cost of a single subproblem somewhere in set of recursive function invocation
- sum the costs within each level of the tree to obtain a set of per-level costs
- sum all the per-level costs to determine the total cost of all levels of the recursion

- 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

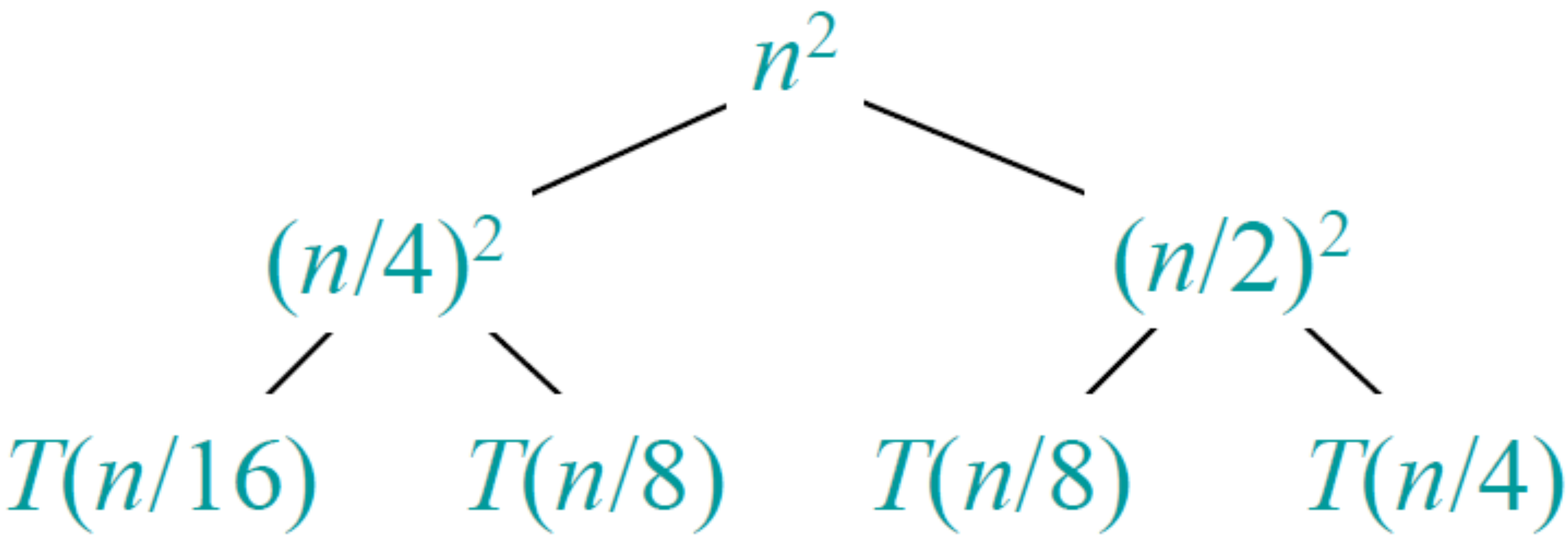
Solve $T(n) = T(n/4) + T(n/2) + n^2$:

$$n^2$$

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

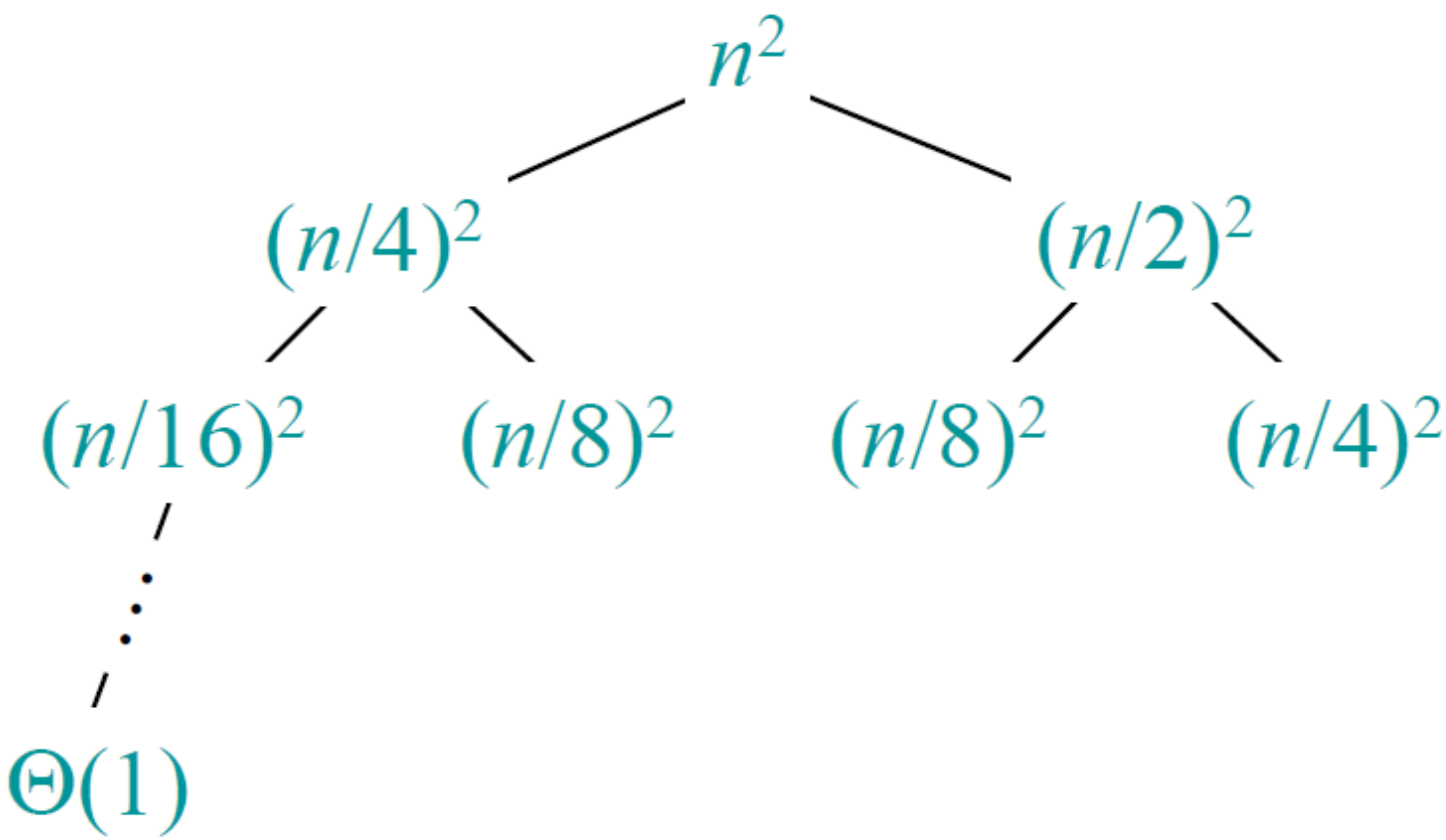
- <u>Example of recursion tree</u>

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



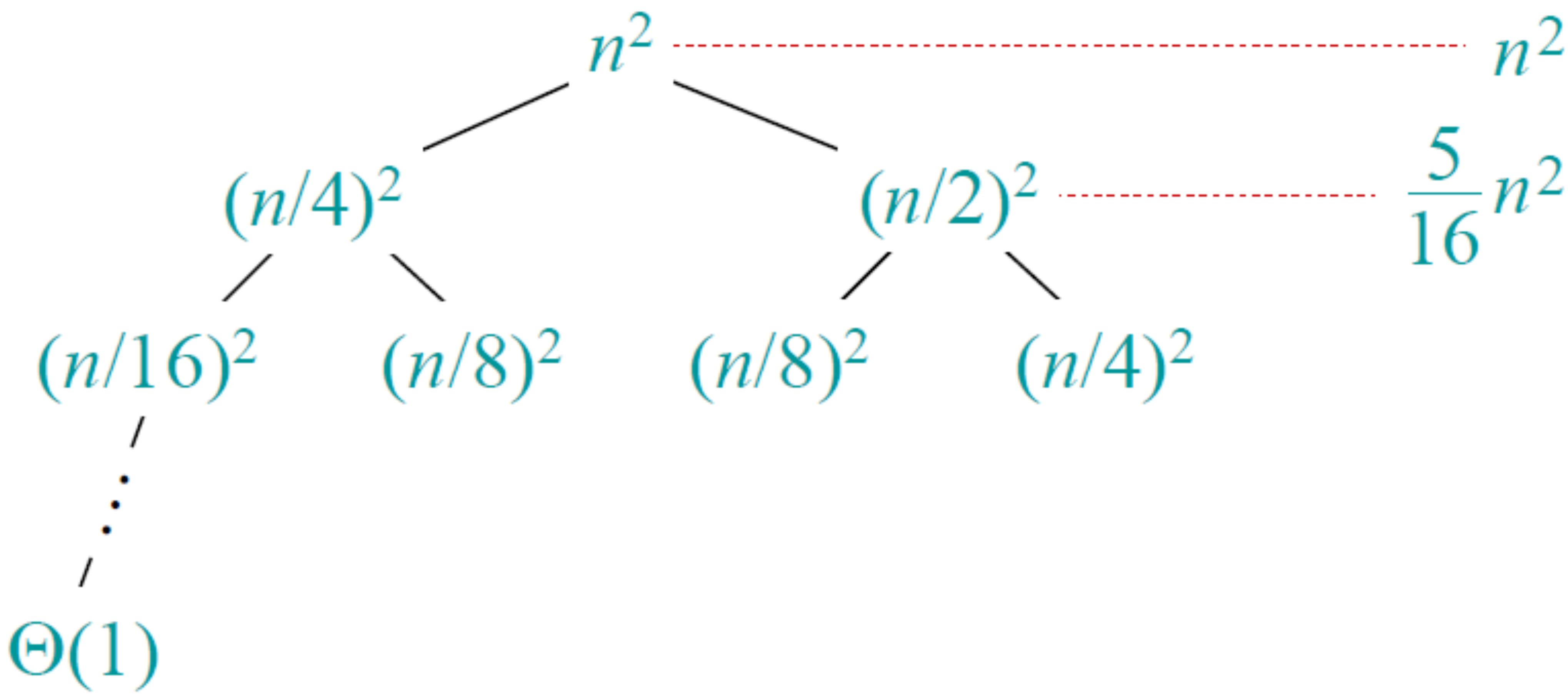
- <u>Example of recursion tree</u>

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



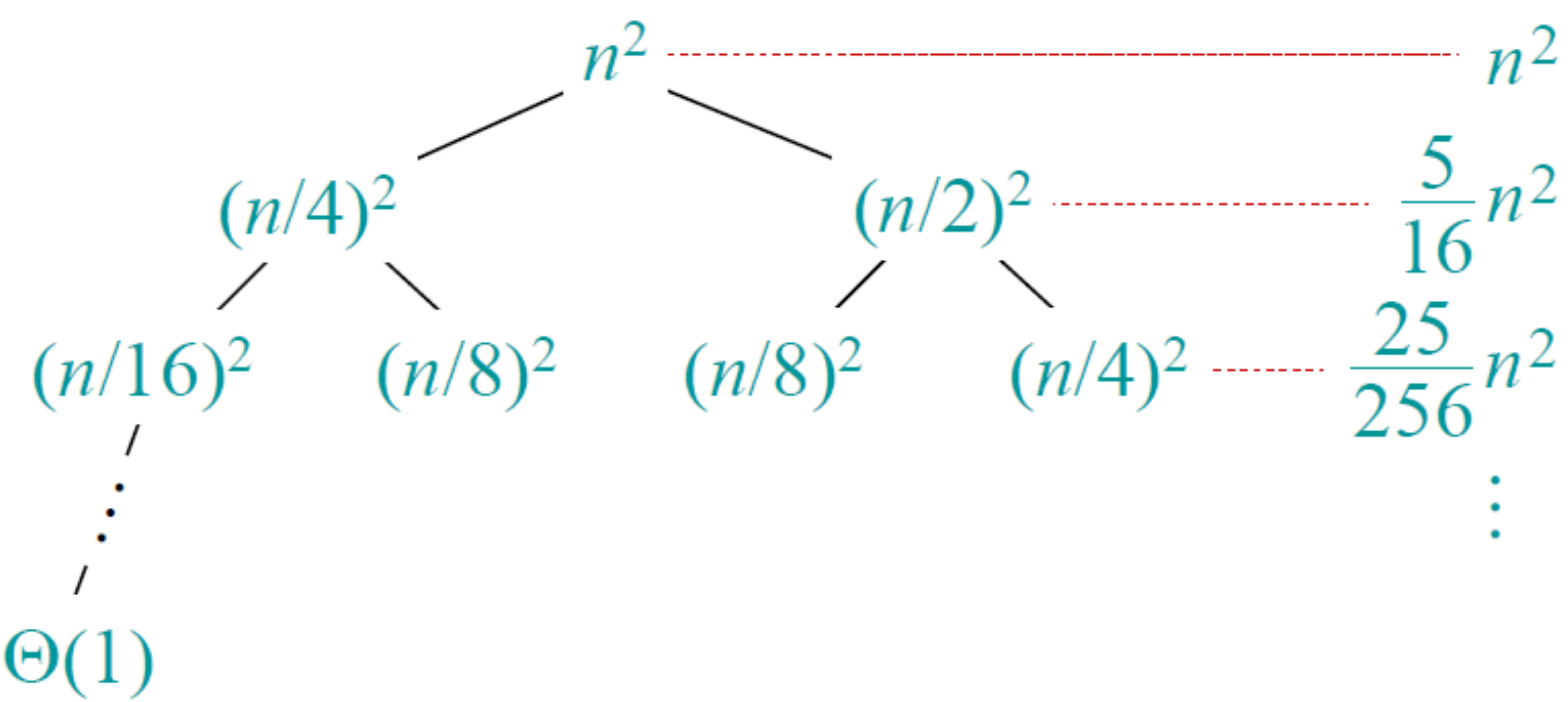
- <u>Example of recursion tree</u>

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



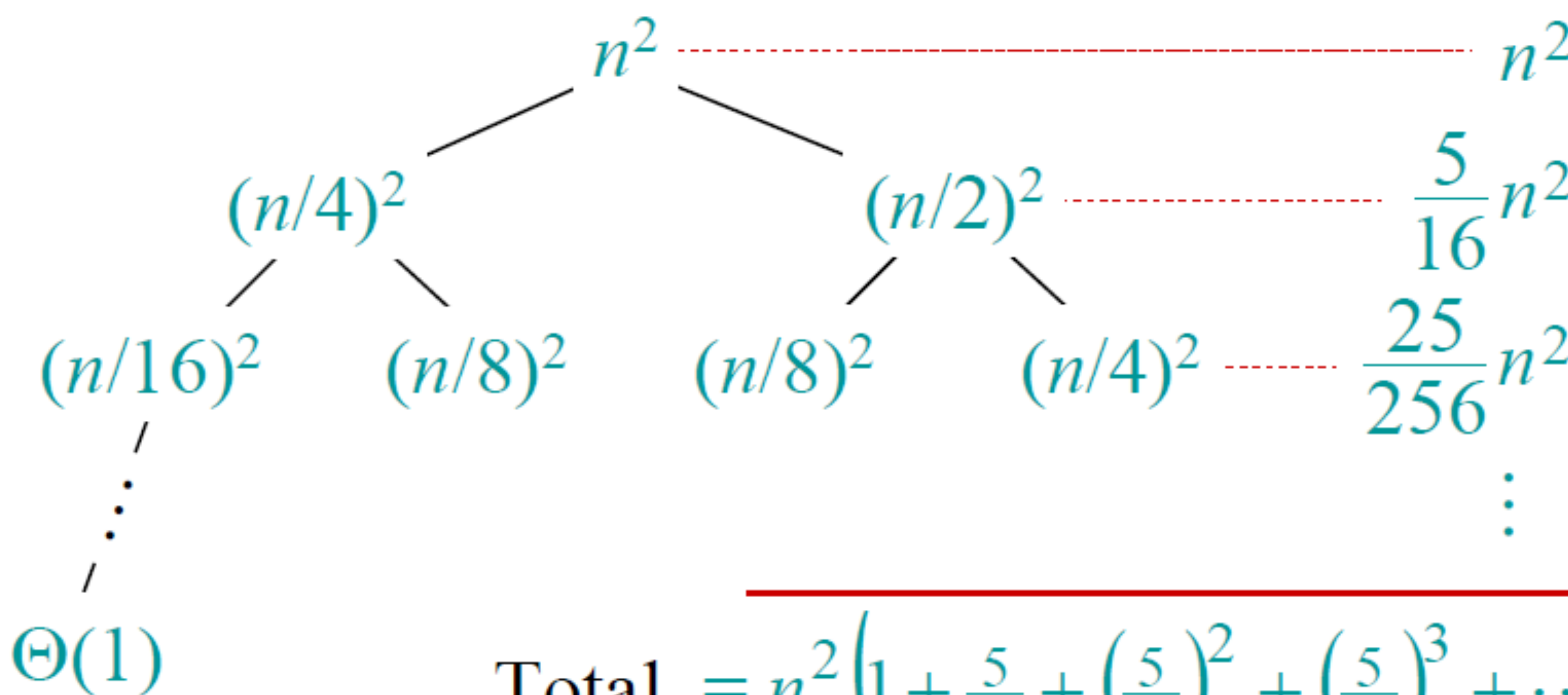
- <u>Example of recursion tree</u>

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



- <u>Example of recursion tree</u>

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



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

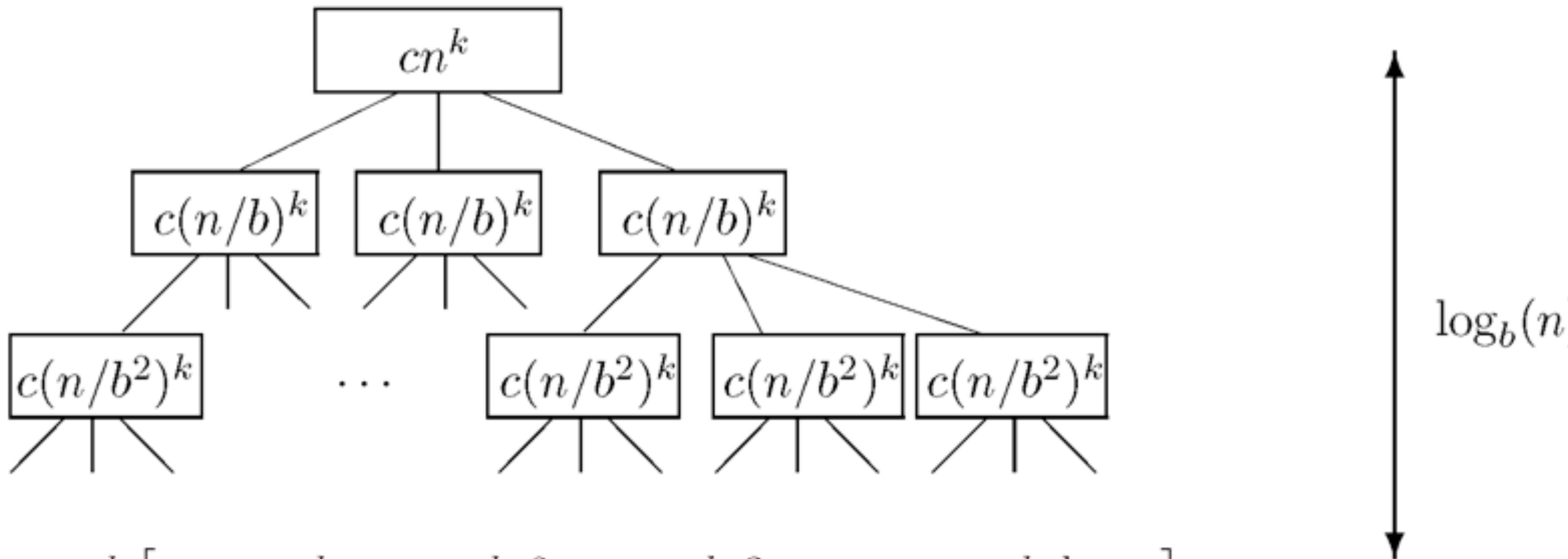
geometric series

master theorem

- The master method applies to recurrences of the form

$$T(n) = aT(n/b) + cn^k$$

- where $a \geq 1$, $b > 1$, and $k \geq 0$ is asymptotically positive.



$$cn^k\left[1 + a/b^k + (a/b^k)^2 + (a/b^k)^3 + \ldots + (a/b^k)^{\log_b n}\right]$$

$$r = a/b^k \qquad cn^k\left[1 + r + r^2 + r^3 + \ldots + r^{\log_b n}\right]$$

Case 1: $r < 1$. $\quad 1 + r + r^2 + \ldots = 1/(1-r)$ $\quad \boxed{T(n) \in \Theta(n^k) \text{ if } a < b^k}$

Case 2: $r = 1$ $\quad \boxed{T(n) \in \Theta(n^k \log n) \text{ if } a = b^k}$

Case 3: $r > 1$ $\quad cn^k r^{\log_b n}\left[(1/r)^{\log_b n} + \ldots + 1/r + 1\right]$

$$\boxed{T(n) \in \Theta(n^{\log_b a}) \text{ if } a > b^k}$$

master theorem

$$
\begin{aligned}
T(n) &= aT(n/b) + cn^k \\
T(1) &= c,
\end{aligned}
$$

$$
\begin{aligned}
T(n) &\in \Theta(n^k) \;\; \textit{if } a < b^k \\
T(n) &\in \Theta(n^k \log n) \;\; \textit{if } a = b^k \\
T(n) &\in \Theta(n^{\log_b a}) \;\; \textit{if } a > b^k
\end{aligned}
$$