Lectures 4-5.
Recurrences

Introduction to Algorithms
Da Nang University of Science and Technology

Dang Thien Binh
dtbinh@dut.udn.vn

Copyright 2000-2022 Intelligent Networking Laboratory

Overview for Recurrences

- Define what a recurrence is
- Discuss three methods of solving recurrences
  - Substitution method
  - Recursion-tree method
  - Master method
- Examples of each method

Definition

- A ***recurrence*** is an equation or inequality that describes a function in terms of its value on smaller inputs.
- Example from MERGE-SORT

$$T(n) = \begin{cases} \Theta(1) & \text{if } n = 1 \\ 2T(n/2) + \Theta(n) & \text{if } n > 1 \end{cases}$$

Technicalities

- Normally, independent variables only assume integral values
- Example from MERGE-SORT revisited

$$T(n) = \begin{cases} \Theta(1) & \text{if } n = 1 \\ T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n) & \text{if } n > 1 \end{cases}$$

- For simplicity, ignore floors and ceilings – often insignificant

Technicalities

- Boundary conditions (small n) are also glossed over

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

- Value of $T(n)$ assumed to be small constant for small n

Substitution Method

- Involves two steps:
  - Guess the form of the solution
  - Use mathematical induction to find the constants and show the solution works
- Drawback
  - Applied only in cases where it is easy to guess at solution
- Useful in estimating bounds on true solution even if latter is unidentified

Substitution Method

- Example:

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

- Guess:

$$T(n) = O(n \lg n)$$

- Prove by induction:

$$T(n) \le cn \lg n$$

for suitable $c > 0$

Inductive Proof

- We'll not worry about the basis case for the moment – we'll choose this as needed – clearly we have:

$$T(1) = \Theta(1) \le cn \lg n \quad \textit{(Page 84, textbook)}$$

- Inductive hypothesis:
  - For values of $n < k$ the inequality holds,
  i.e., $T(n) \le cn \lg n$
  - We need to show that this holds for $n = k$ as well.

Inductive Proof

- In particular, for $n = \lfloor k/2 \rfloor$, the inductive hypothesis should hold, *i.e.*,

  $$T(\lfloor k/2 \rfloor) \le c \lfloor k/2 \rfloor \lg \lfloor k/2 \rfloor$$

  The recurrence gives us:

  $$T(k) = 2T(\lfloor k/2 \rfloor) + k$$

  Substituting the inequality above yields:

  $$T(k) \le 2[c \lfloor k/2 \rfloor \lg \lfloor k/2 \rfloor] + k$$

Inductive Proof

- Because of the non-decreasing nature of the functions involved, we can drop the "floors" and obtain:

  $$T(k) \le 2[c\,(k/2) \lg (k/2)] + k$$

  Which simplifies to:

  $$T(k) \le ck\,(\lg k - \lg 2) + k$$

  Or, since $\lg 2 = 1$, we have:

  $$T(k) \le ck \lg k - ck + k = ck \lg k + (1 - c)k$$

  So if $c \ge 1$, $T(k) \le ck \lg k$ ***Q.E.D.***

Practice Problems

- Use inductive proof to show that $T(n) \in O(n^2)$

$$\begin{cases} T(n) = 7T(n/3) + n^2 \\ T(1) = 1 \end{cases}$$

Will be solved in Q&A session

Recursion-Tree Method

- Straightforward technique of coming up with a good guess
- Can help the Substitution Method
- ***Recursion tree***: visual representation of recursive call hierarchy where <u>each node represents the cost of a single subproblem</u>

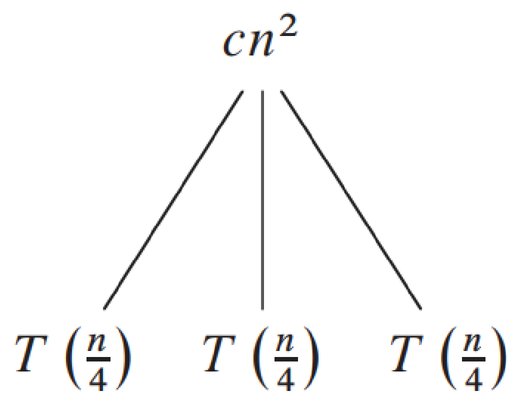
Recursion-Tree Method

$$T(n) = 3T(\lfloor n/4 \rfloor) + \Theta(n^2)$$

$$\boxed{T(n)}$$

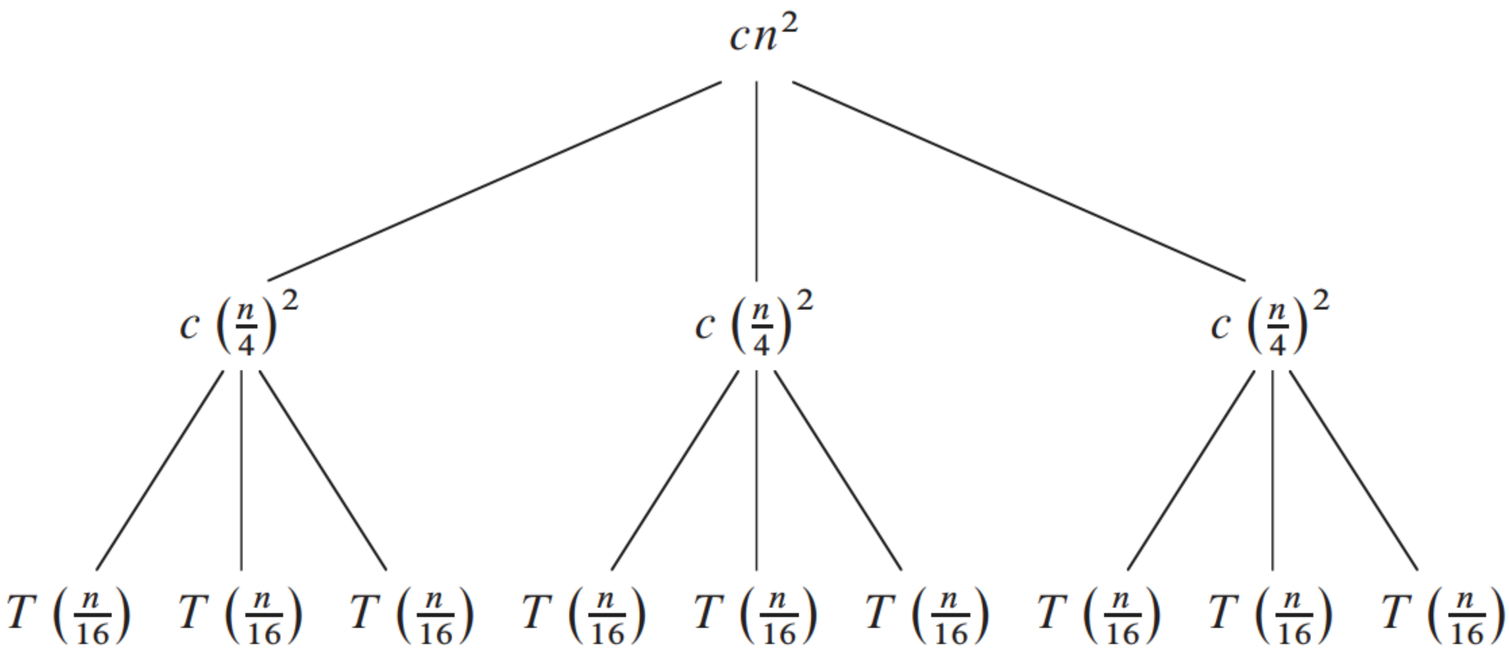
Recursion-Tree Method

$$T(n) = 3T(\lfloor n/4 \rfloor) + \Theta(n^2)$$



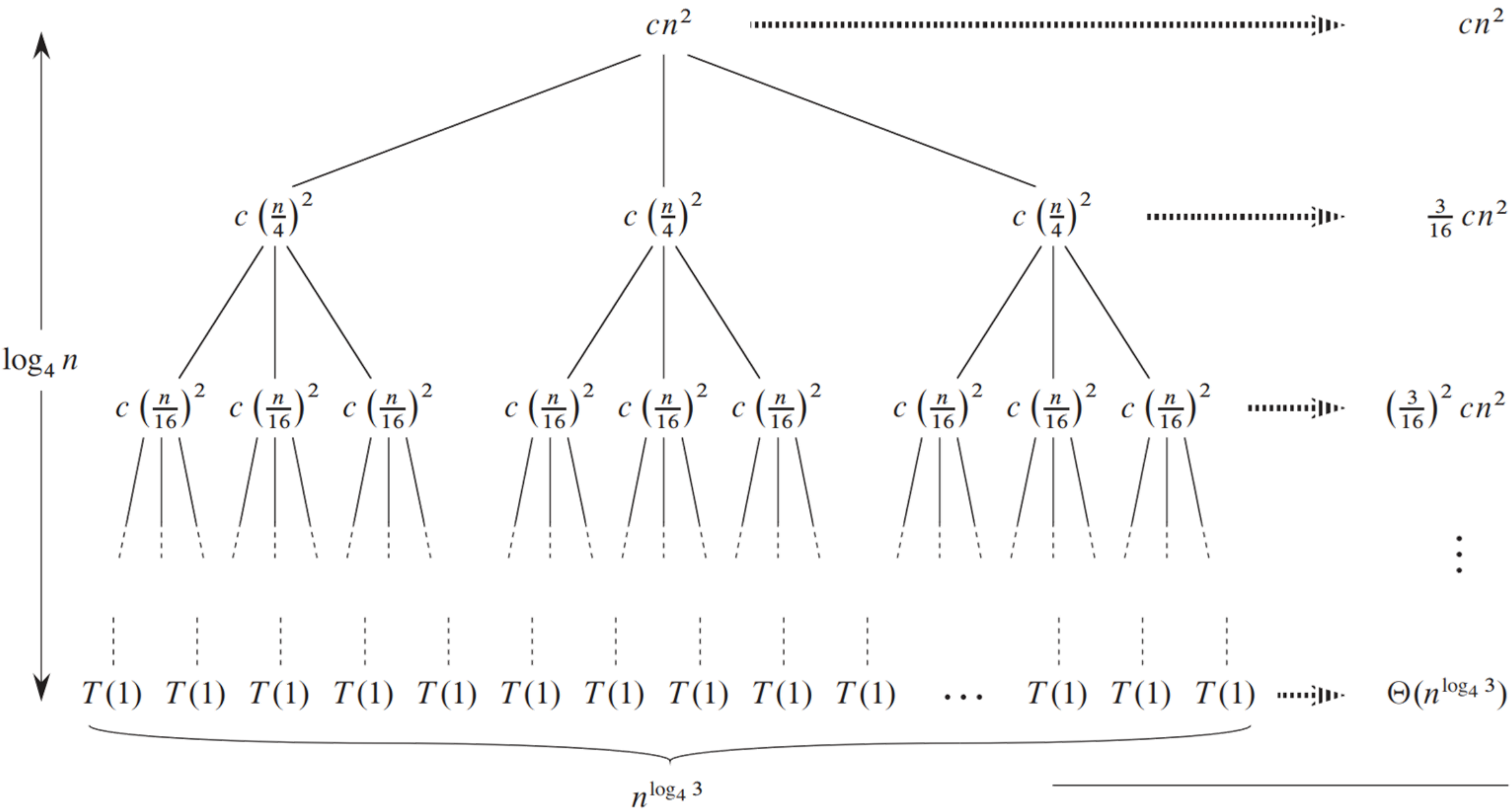
Recursion-Tree Method

$$T(n) = 3T(\lfloor n/4 \rfloor) + \Theta(n^2)$$



Recursion-Tree Method

Page 90, textbook



Recursion-Tree Method

- Gathering all the costs together:

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

$$T(n) \le \sum_{i=0}^{\infty} (3/16)^i cn^2 + o(n)$$

$$T(n) \le (1/(1-3/16))cn^2 + o(n)$$

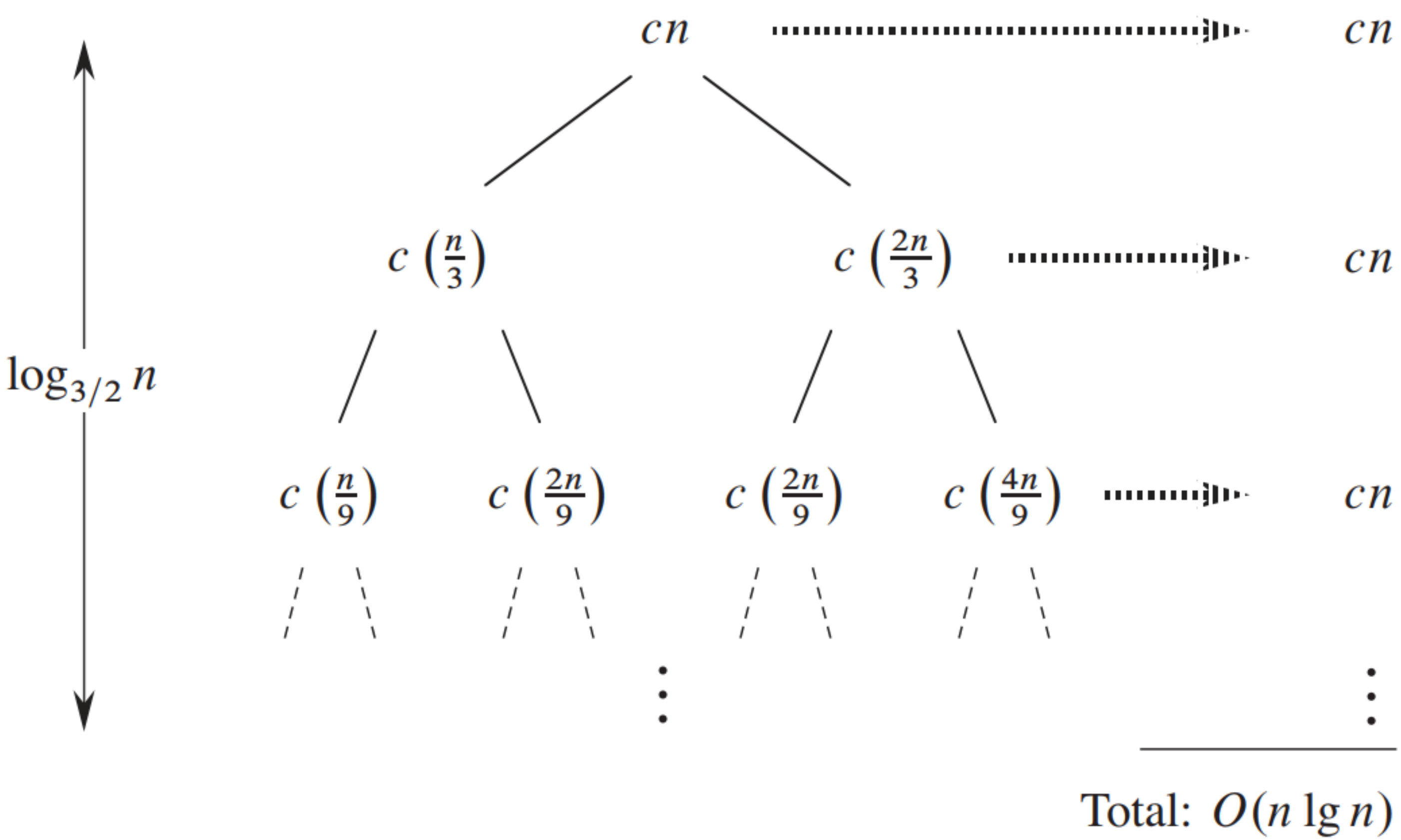
$$T(n) \le (16/13)cn^2 + o(n)$$

$$T(n) = O(n^2)$$

Recursion-Tree Method

Page 92, textbook

$$T(n) = T(n/3) + T(2n/3) + O(n)$$



Recursion-Tree Method

- An overestimate of the total cost:

$$T(n) = \sum_{i=0}^{\log_{3/2} n - 1} cn + \Theta(n^{\log_{3/2} 2})$$

- Counter-indications:

$$T(n) = O(n \lg n) + \omega(n \lg n)$$

- Notwithstanding this, use as “guess”:

$$T(n) = O(n \lg n)$$

Substitution Method

- Recurrence:

$$T(n) = T(n/3) + T(2n/3) + cn$$

- Guess:

$$T(n) = O(n \lg n)$$

- Prove by induction:

$$T(n) \le dn \lg n$$

for suitable $d > 0$ (we already use c)

Inductive Proof

- Again, we'll not worry about the basis case
- Inductive hypothesis:
  For values of $n < k$ the inequality holds,
  i.e., $T(n) \le dn \lg n$
  We need to show that this holds for $n = k$ as well
- In particular, for $n = k/3$, and $n = 2k/3$, the inductive hypothesis should hold…

Inductive Proof

- That is

$$T(k/3) \le d\ k/3 \lg k/3$$
$$T(2k/3) \le d\ 2k/3 \lg 2k/3$$

- The recurrence gives us:

$$T(k) = T(k/3) + T(2k/3) + ck$$

- Substituting the inequalities above yields:

$$T(k) \le [d\ (k/3) \lg (k/3)] + [d\ (2k/3) \lg (2k/3)] + ck$$

Inductive Proof

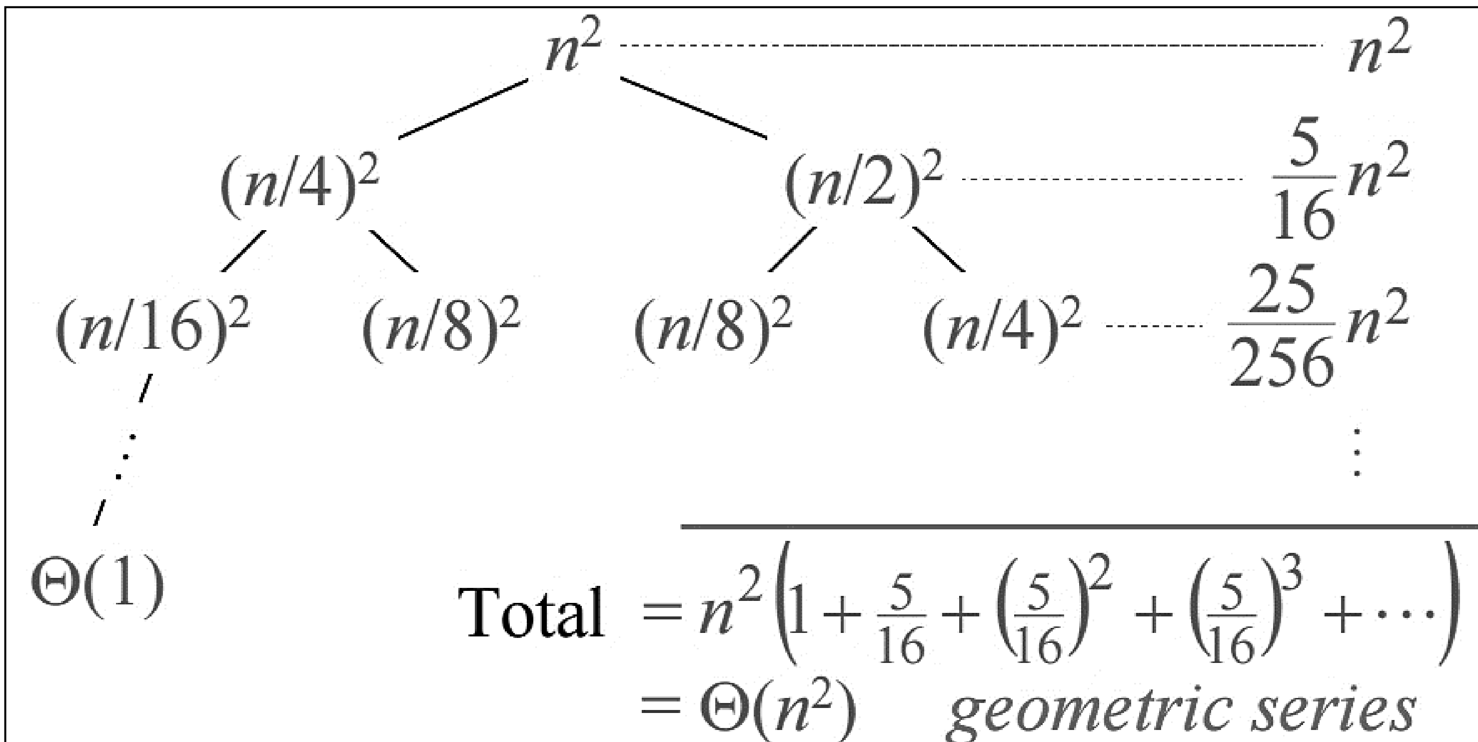
- Expanding, we get
  $T(k) \le [d\,(k/3)\lg k - d\,(k/3)\lg 3] +$
  $[d\,(2k/3)\lg k - d\,(2k/3)\lg(3/2)] + ck$
- Rearranging, we get:
  $T(k) \le dk \lg k - d[(k/3)\lg 3 + (2k/3)\lg(3/2)] + ck$
  $T(k) \le dk \lg k - dk[\lg 3 - 2/3] + ck$
- When $d \ge c/(\lg 3 - (2/3))$, we should have the desired:

$$T(k) \le dk \lg k$$

Practice Problems

- Use the recursion tree method to show that $T(n) \in \Theta(n^2)$

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



Master Method

- Provides a "cookbook" method for solving recurrences
- Recurrence must be of the form:

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

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

Master Method

- ***Theorem 4.1:***

  Given the recurrence previously defined, we have:

  1. If $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$,
     then $T(n) = \Theta(n^{\log_b a})$
  2. If $f(n) = \Theta(n^{\log_b a})$,
     then $T(n) = \Theta(n^{\log_b a} \lg n)$
  3. If $f(n) = \Omega(n^{\log_b a + \varepsilon})$ for some constant $\varepsilon > 0$,
     and if $af(n/b) \le cf(n)$ for some constant $c < 1$
     and all sufficiently large n,
     then $T(n) = \Theta(f(n))$

Example

- Estimate bounds on the following recurrence:

$$T(n) = \begin{cases} 1 & \text{if } n = 1, \\ 3T(n/2) + n^2 & \text{otherwise.} \end{cases}$$

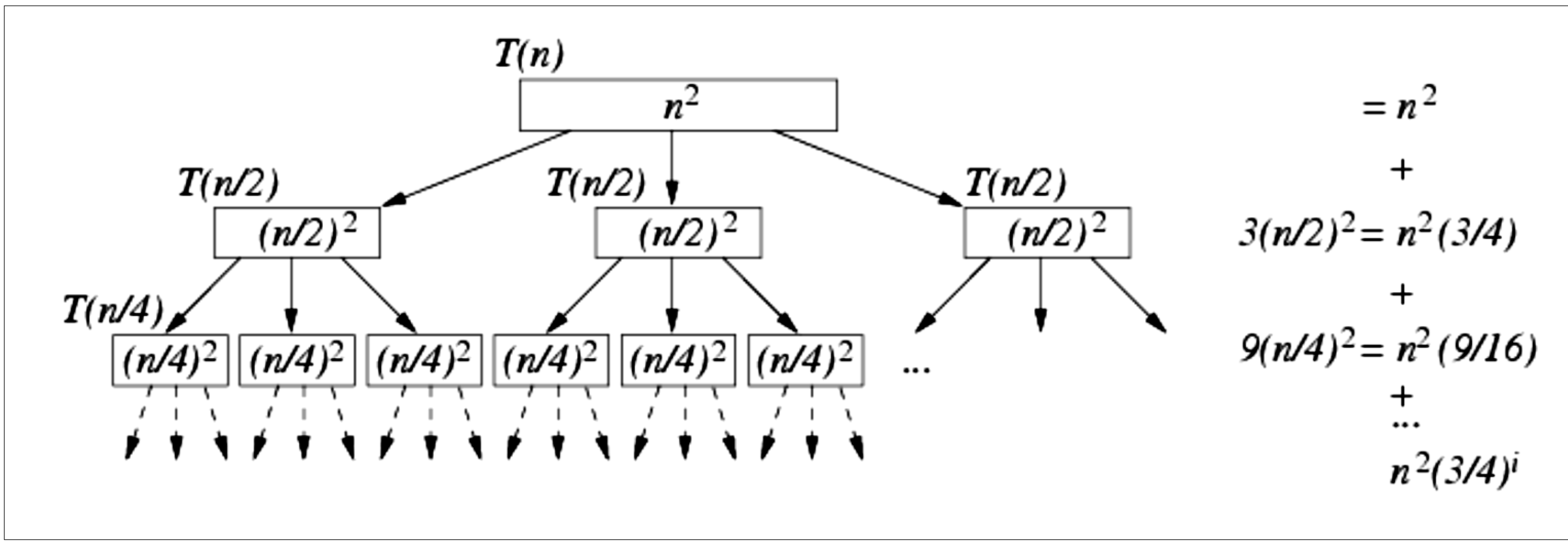
- Use the recursion tree method to arrive at a "guess" then verify using induction

- Point out which case in the Master Method this falls in

Recursion Tree

$$T(n) = \begin{cases} 1 & \text{if } n = 1, \\ 3T(n/2) + n^2 & \text{otherwise.} \end{cases}$$

- Recurrence produces the following tree:



Cost Summation

- Collecting the level-by-level costs:

$$T(n) = n^2 \sum_{i=0}^{?} \left(\frac{3}{4} \right)^i$$

- A geometric series with base less than one; converges to a finite sum, hence, $T(n) = \Theta(n^2)$

Exact Calculation

- If an exact solution is preferred:

$$T(n) = n^2 \sum_{i=0}^{\lg n} \left(\frac{3}{4}\right)^i$$

- Using the formula for a partial geometric series:

$$T(n) = n^2 \frac{(3/4)^{\lg n+1} - 1}{(3/4) - 1}$$

Exact Calculation

- Solving further:

$$T(n) = n^2 \frac{(3/4)^{\lg n+1} - 1}{(3/4) - 1} = -4n^2((3/4)^{\lg n+1} - 1)$$

$$= 4n^2(1 - (3/4)^{\lg n+1}) = 4n^2(1 - (3/4)(3/4)^{\lg n})$$

$$= 4n^2(1 - (3/4)n^{\lg(3/4)}) = 4n^2(1 - (3/4)n^{\lg 3 - \lg 4})$$

$$= 4n^2(1 - (3/4)n^{\lg 3 - 2)}) = 4n^2(1 - (3/4)(n^{\lg 3} / n^2))$$

$$= 4n^2 - 3n^{\lg 3}$$

Master Theorem (Simplified)

Theorem: (Simplified Master Theorem) Let $a \geq 1$, $b > 1$ be constants and let $T(n)$ be the recurrence

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

defined for $n \geq 0$. (As usual let us assume that n is a power of b. The basis case, $T(1)$ can be any constant value.) Then

Case 1: if $a > b^k$ then $T(n) \in \Theta(n^{\log_b a})$.

Case 2: if $a = b^k$ then $T(n) \in \Theta(n^k \log n)$.

Case 3: if $a < b^k$ then $T(n) \in \Theta(n^k)$.

$$T(n) = \begin{cases} 1 & \text{if } n = 1, \\ 3T(n/2) + n^2 & \text{otherwise.} \end{cases}$$

Practice Problems

- Use Master theorem to find asymptotic bound
  a. $T(n) = 4T(n/2) + n$
  b. $T(n) = 4T(n/2) + n^2$
  c. $T(n) = 4T(n/2) + n^3$

  Will be solved in Q&A session

Thanks to contributors

Mr. Phuoc-Nguyen Bui (2022)

Dr. Thien-Binh Dang (2017 – 2022)

Prof. Hyunseung Choo (2001 – 2022)

Copyright 2000-2022 Intelligent Networking Laboratory