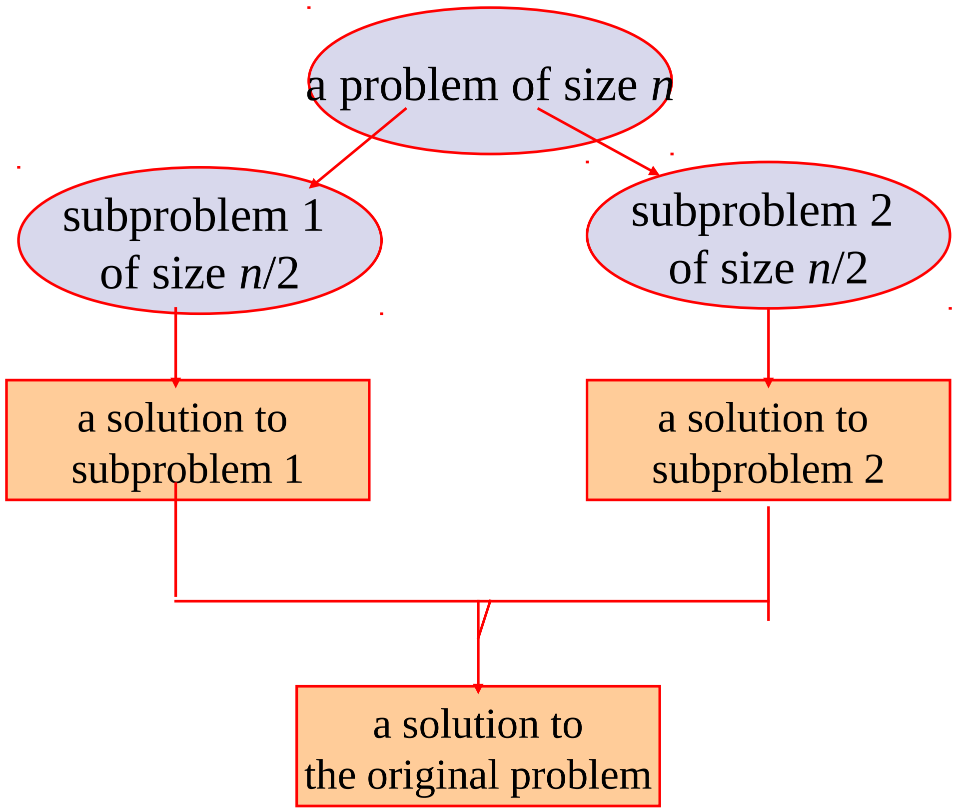
COMP20003
Algorithms and Data Structures
Recurrences

Nir Lipovetzky
Department of Computing and Information Systems
University of Melbourne
Semester 2

Divide and Conquer Algorithms

- Mergesort and quicksort are instances of divide-and-conquer algorithms:
  - Solve the problem by continually dividing into smaller problems.
- Other examples?

Split-solve-join approach:

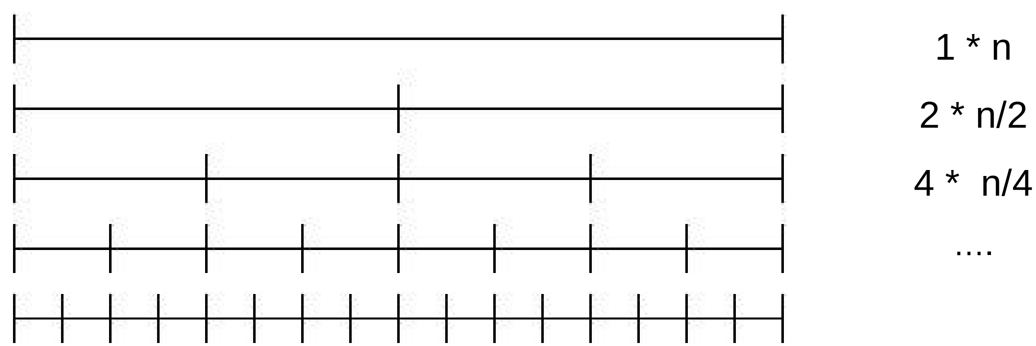


For problems where the output is a transformation of the input, need to:

- process both sub-problems, and
- join the sub-solutions after processing

Recurrence for divide and conquer sorting algorithms

- **One pass** through the data reduces problem size by half. Process both halves.
- Operation takes constant time *c*.
- Base case takes time *d*.



Recurrence for divide and conquer sorting algorithms

- **One pass** through the data reduces problem size by half. Process both halves.
- Operation takes constant time *c.*
- Base case takes time *d.*

$$T(1) = d$$

$$T(n) = 2T(n/2) + nc$$

$$= nc + 2cn/2 + 4cn/4 \ldots + n/2*2c + nd$$

$$= c(n-1)\log n + nd$$

Divide and Conquer: Recurrences to Master Theorem

- Most common case:

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

- General case:

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

$$f(n) \in \Theta(n^d)$$

- Most common case:

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

$$a=2,\ b=2,\ d=1$$

Master Theorem for Divide and Conquer

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

  $$f(n) \in \Theta(n^d)$$

- *T(n) closed form varies, depending on whether:*
  - $d > \log_b a \qquad T(n) \in \Theta(n^d)$
  - $d = \log_b a \qquad T(n) \in \Theta(n^d \log n)$
  - $d < \log_b a \qquad T(n) \in \Theta(n^{\log_b a})$

Master Theorem for Divide and Conquer

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

  $a \geq 1$, $b > 1$, n^d asymptotically positive

- *$T(n)$ closed form varies, depending on whether:*
  - $d > \log_b a$ $\quad T(n) \in \Theta(n^d)$
  - $d = \log_b a$ $\quad T(n) \in \Theta(n^d \log n)$
  - $d < \log_b a$ $\quad T(n) \in \Theta(n^{\log_b a})$

Where do Θ() solutions to the Master Theorem come from?

$T(n) = aT(n/b) + f(n), \ f(n) \in \Theta(n^d)$

- Size of subproblems decreases by b
  - So base case reached after $log_b n$ levels
  - Recursion tree $log_b n$ levels
- Branch factor is a
  - At kth level, have a^k subproblems
- At level k, total work is then
  - $a^k * O(n/b^k)^d$
  - *(#subproblems * cost of solving one)*

Where do Θ() solutions to the Master Theorem come from?

$T(n) = aT(n/b) + f(n)$, $f(n) \in \Theta(n^d)$

- At level k, total work is then
  - $a^k * O(n/b^k)^d = O(n^d) * (a/b^d)^k$
- As k (levels) goes from 0 to $\log_b n$, this is a geometric series, with ratio a/b^d.

$$\Sigma\ O(n^d) * (a/b^d)^k$$

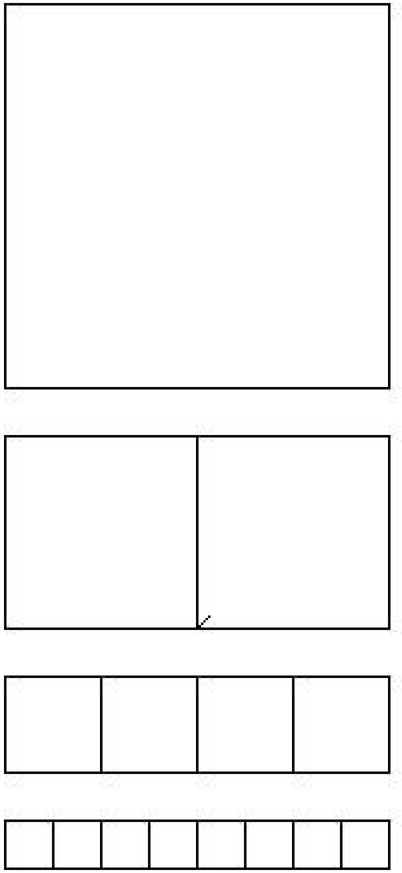
Where do Θ() solutions to the Master Theorem come from?

$T(n) = aT(n/b) + f(n)$, $f(n) \in \Theta(n^d)$

- *Geometric series:* $O(n^d) * (a/b^d)^k$
  - as k goes from $0 \rightarrow \log_b n$
- Case 1: *ratio* $a/b^d < 1$
  - $(a/b^d)^k$ *gets smaller as* k *goes from* $1 \rightarrow \log n$
  - a/b^d First term is the largest, and is <1
  - $O(n^d)$

Example for $a/b^d < 1$

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



Where do the solutions to the Master Theorem come from?

$T(n) = aT(n/b) + f(n)$, $f(n) \in \Theta(n^d)$

- *Geometric series:* $O(n^d) * (a/b^d)^k$
  - as k goes from $0 \rightarrow \log_b n$
- Case 2: *ratio* $a/b^d = 1$
  - Series is $O(n^d) + O(n^d) + \ldots$
    - For $\log_b n$ levels
  - Sum $= O(n^d \log n)$

Example for most common case $a/b^d = 1$

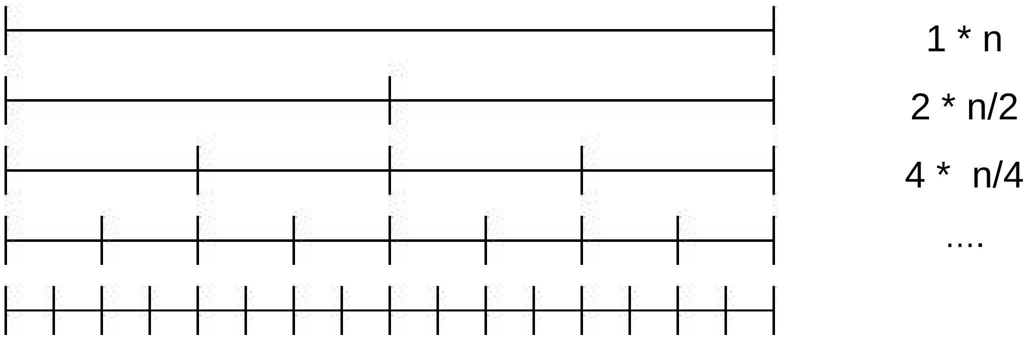
$$T(n) = 2T(n/2) + n$$

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

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

$$= 8T(n/8) + n + n + n$$

….



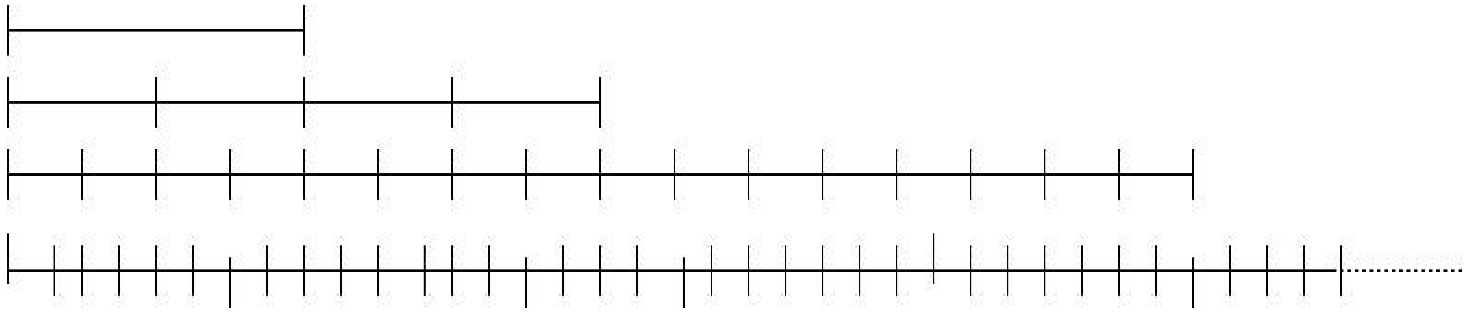
Where do Θ() solutions to the Master Theorem come from?

$T(n) = aT(n/b) + f(n)$, $f(n) \in \Theta(n^d)$

- *Geometric series:* $O(n^d) * (a/b^d)^k$
  - as k goes from $0 \rightarrow \log_b n$
- *Case 3: ratio* $a/b^d > 1$
  - $a/b^d > 1 \rightarrow$ series is *increasing*
  - Sum dominated by last term:
    - $O(n^d)(a/b^d)^{\log(b)n} = n^{\log(b)a}$

Example for $a/b^d > 1$

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



- For more on geometric series, and calculation of closed form, see: http://www.youtube.com/watch?v=JJZ-shHiayU

- 4 minute tutorial from Rose-Hulman Institute of Technology