CS 350 Algorithms and Complexity

Lecture 09
Divide & Conquer, Master Theorem

Paul Doliotis – Adjunct Assistant Professor
Portland State University

What is Divide-and-Conquer?

Solves a problem instance of size n by:

1. dividing it into b smaller instances, of size ~ n/b

What is Divide-and-Conquer?

Solves a problem instance of size n by:

1. dividing it into b smaller instances, of size ~ n/b
2. solving some or all of them (in general, solving a of them), using the same algorithm recursively.

What is Divide-and-Conquer?

Solves a problem instance of size n by:

1. dividing it into b smaller instances, of size ~ n/b
2. solving some or all of them (in general, solving a of them), using the same algorithm recursively.
3. combining the solutions to the a smaller problems to get the solution to the original problem.

What is Divide-and-Conquer?

Solves a problem instance of size n by:

1. dividing it into b smaller instances, of size ~ n/b
2. solving some or all of them (in general, solving a of them), using the same algorithm recursively.
3. combining the solutions to the a smaller problems to get the solution to the original problem.

Time taken?

What is Divide-and-Conquer?

Solves a problem instance of size n by:

1. dividing it into b smaller instances, of size ~ n/b
2. solving some or all of them (in general, solving a of them), using the same algorithm recursively.
3. combining the solutions to the a smaller problems to get the solution to the original problem.

Time taken? T(n) = a*T(n/b)   true or false

What is Divide-and-Conquer?

Solves a problem instance of size n by:

1. dividing it into b smaller instances, of size ~ n/b
2. solving some or all of them (in general, solving a of them), using the same algorithm recursively.
3. combining the solutions to the a smaller problems to get the solution to the original problem.

$$T(n) = a*T(n/b) + T_{\text{split and combine}}(n)$$

What is Divide-and-Conquer?

Divide-and-Conquer

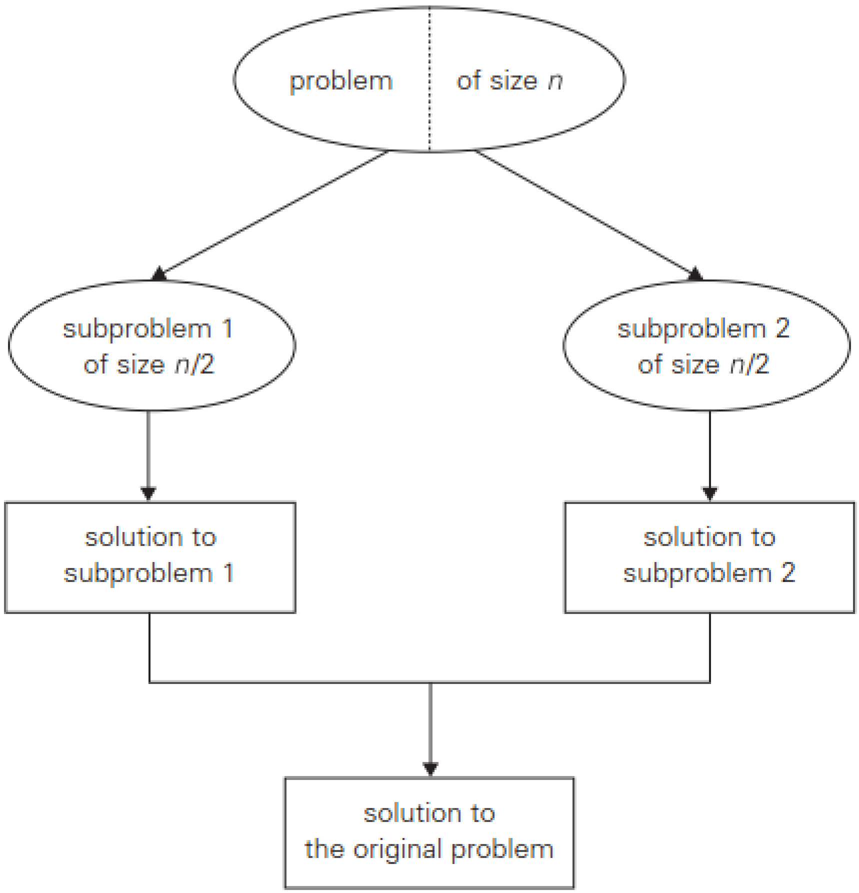

FIGURE 5.1 Divide-and-conquer technique (typical case).

Example

- Lets compute the sum of n elements with divide and conquer
- $a_0, a_1, a_2, \ldots, a_{n-1}, n>1$

Example

- Lets compute the sum of n elements with divide and conquer
- a_0, a_1, a_2,…,a_{n-1}, n>1

$$a_0 + \cdots + a_{n-1} = (a_0 + \cdots + a_{\lfloor n/2 \rfloor - 1}) + (a_{\lfloor n/2 \rfloor} + \cdots + a_{n-1}).$$

Example

- Lets compute the sum of n elements with divide and conquer
- a_0, a_1, a_2,…,a_{n-1}, n>1

$$a_0 + \cdots + a_{n-1} = (a_0 + \cdots + a_{\lfloor n/2 \rfloor - 1}) + (a_{\lfloor n/2 \rfloor} + \cdots + a_{n-1}).$$

What is the recursive formula for number of additions?

Example

- Lets compute the sum of n elements with divide and conquer
- a_0, a_1, a_2,…,a_{n-1}, n>1

$$a_0 + \cdots + a_{n-1} = (a_0 + \cdots + a_{\lfloor n/2 \rfloor - 1}) + (a_{\lfloor n/2 \rfloor} + \cdots + a_{n-1}).$$

What is the recursive formula for number of additions?

A(n) = 2*A(n/2) + 1, A(1) = 0, n>1

Example

- Lets compute the sum of n elements with divide and conquer
- $a_0, a_1, a_2, \ldots, a_{n-1}$, n>1

$$a_0 + \cdots + a_{n-1} = (a_0 + \cdots + a_{\lfloor n/2 \rfloor - 1}) + (a_{\lfloor n/2 \rfloor} + \cdots + a_{n-1}).$$

What is the recursive formula for number of additions?

$A(n) = 2*A(n/2) + 1$, $A(1) = 0$, $n>1$
$T(n) = a*T(n/b) + T_{\text{split and combine}}(n)$

Master Theorem

Master Theorem If $f(n) \in \Theta(n^d)$ where $d \geq 0$ in recurrence (5.1), then

$$T(n) \in \begin{cases} \Theta(n^d) & \text{if } a < b^d, \\ \Theta(n^d \log n) & \text{if } a = b^d, \\ \Theta(n^{\log_b a}) & \text{if } a > b^d. \end{cases}$$

Analogous results hold for the O and Ω notations, too.

Master Theorem

Master Theorem If $f(n) \in \Theta(n^d)$ where $d \geq 0$ in recurrence (5.1), then

$$T(n) \in \begin{cases} \Theta(n^d) & \text{if } a < b^d, \\ \Theta(n^d \log n) & \text{if } a = b^d, \\ \Theta(n^{\log_b a}) & \text{if } a > b^d. \end{cases}$$

Analogous results hold for the O and Ω notations, too.

Lets apply Master Theorem to the previous example:

Master Theorem

Master Theorem If $f(n) \in \Theta(n^d)$ where $d \geq 0$ in recurrence (5.1), then

$$T(n) \in \begin{cases} \Theta(n^d) & \text{if } a < b^d, \\ \Theta(n^d \log n) & \text{if } a = b^d, \\ \Theta(n^{\log_b a}) & \text{if } a > b^d. \end{cases}$$

Analogous results hold for the O and Ω notations, too.

Lets apply Master Theorem to the previous example:
A(n) = 2*A(n/2) + 1, A(1) = 0, n>1
T(n) = a*T(n/b) + $T_{\text{split and combine}}$(n)

Master Theorem

Master Theorem If $f(n) \in \Theta(n^d)$ where $d \geq 0$ in recurrence (5.1), then

$$T(n) \in \begin{cases} \Theta(n^d) & \text{if } a < b^d, \\ \Theta(n^d \log n) & \text{if } a = b^d, \\ \Theta(n^{\log_b a}) & \text{if } a > b^d. \end{cases}$$

Analogous results hold for the O and Ω notations, too.

Lets apply Master Theorem to the previous example:

$A(n) = 2*A(n/2) + 1$, $A(1) = 0$, $n>1$

$T(n) = a*T(n/b) + T_{\text{split and combine}}(n)$

$a=2$, $b=2$, $d=0$ $a<b^d$

Master Theorem

Master Theorem If $f(n) \in \Theta(n^d)$ where $d \geq 0$ in recurrence (5.1), then

$$T(n) \in \begin{cases} \Theta(n^d) & \text{if } a < b^d, \\ \Theta(n^d \log n) & \text{if } a = b^d, \\ \Theta(n^{\log_b a}) & \text{if } a > b^d. \end{cases}$$

Analogous results hold for the O and Ω notations, too.

Lets apply Master Theorem to the previous example:
A(n) = 2*A(n/2) + 1, A(1) = 0, n>1
$T(n) = a*T(n/b) + T_{\text{split and combine}}(n)$
a=2, b=2, d=0 $a<b^d$
Θ(n)