Analysis of Algorithms

V. Adamchik CSCI 570 Fall 2016

Discussion 7 University of Southern California

# Divide-And-Conquer

## Problem 1

There are 2 sorted arrays A and B of size n each. Design a D&C algorithm to find the median of the array obtained after merging the above 2 arrays (i.e. array of length 2n). Discuss its runtime complexity.

A = [1, 3, 5, 16, 18, 21, 30]

B = [2, 13, 17, 20, 23, 29, 35]

A∪B = [1,2,3,5,13,16,17,18,20,21,23,29,30,35]

The median: (17+18)/2

## NOT D&C Solutions

1. Merge two sorted arrays. O(n)
2. Sort it. O(n log n)
3. Get the median. O(1)

1. Merge two sorted arrays into a new sorted array. O(n)
2. Get the median. O(1)

## Solution

A = [1, 3, 5, 16, 18, 21, 30]

B = [2, 13, 17, 20, 23, 29, 35]

Compare two medians: $m_A$ = 16 and $m_B$ =20

Since $m_A$ < $m_B$ we can discard some elements

A = [~~1, 3, 5, 16~~, 18, 21, 30]

B = [2, 13, 17, ~~20, 23, 29, 35~~]

to get

A' = [18, 21, 30]

B' = [2, 13, 17]

## Solution

A' = [18, 21, 30]

B' = [2, 13, 17]

Compare two medians. Since 21 > 13

we discard

A' = [18, ~~21, 30~~]

B' = [~~2, 13~~, 17]

to get

A'' = [18]

B'' = [17]

Stop and return the median (18+17)/2

## Solution

Runtime complexity.

Let T(n) denotes the solution to the problem of size n.

Dividing step: during the algorithm execution, in each step, we eliminate half of the elements.

Conquering step: in each step, we compare two medians.

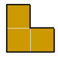
T(n) = T(n/2) + O(1)

T(1) = O(1)

T(n) = O(log n)

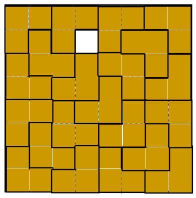
## Problem 2

A tromino is a figure composed of three 1x1 squares in the shape of an L.



Given a $2^n x 2^n$ checkerboard with 1 missing square, tile it with L-trominoes.

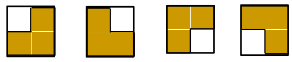
Design a D&C algorithm and discuss its runtime complexity.



## Solution

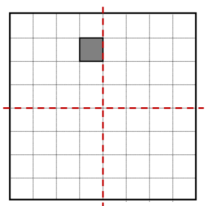
Idea – reduce the size of the original problem, so that we eventually get to the 2x2 boards which we know how to solve…

Tiling a 2x2 board:



## Solution

Let's divide the original board into four boards

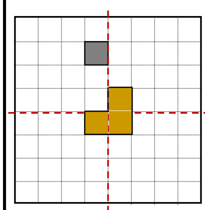


We have one problem of the size $2^{n-1} x 2^{n-1}$

The other three do not have holes!

## Solution

Insert one tromino at the center



Now we have four boards with holes of the size $2^{n-1} x 2^{n-1}$.

Keep doing this division, until we get the 2x2 boards

## Solution

Runtime complexity.

Let T(n) denotes the solution to the problem of size n, where n is a power of 2.

Dividing step: during the algorithm execution, we split the problem into four subproblems of size n/2 each.

Conquering step: in each step, we put a tile in the middle

$T(n) = 4\,T(n/2) + O(1)$    $T(n) = O(n^2)$

$T(1) = O(1)$

## Problem 3

The standard multiplication of two n-digit integers involves $n^2$ single digit multiplications.

```
   1234
  x1111
   1234
  1234
 1234
+1234
1370974
```

Design a D&C algorithm to multiply two n-digit integers. Discuss its runtime complexity.

## Solution

We can split an n-digit integer into two n/2-digit integers. For example,

$$154517766 = 15451 \cdot 10^4 + 7766$$

Generally,

$$num = x_1 \cdot 10^{n/2} + x_0$$

Thus, the product of two integers is

$$(x_1 \cdot 10^{n/2} + x_0) \cdot (y_1 \cdot 10^{n/2} + y_0)$$

After expanding

$$x_1 \cdot y_1 \cdot 10^n + (x_0 \cdot y_1 + x_1 \cdot y_0) \cdot 10^{n/2} + x_0 \cdot y_0$$

## Solution

$$x_1 \cdot y_1 \cdot 10^n + (x_0 \cdot y_1 + x_1 \cdot y_0) \cdot 10^{n/2} + x_0 \cdot y_0$$

Multiplication of two n-digit integers has been reduced to 4 multiplications of n/2-digit integers.

We do not count multiplication by 10, since its complexity is $O(1)$. Additions are $O(1)$ as well.

Let $T(n)$ be a runtime complexity of multiplication of two n-digit integers, then

$T(n) = 4\, T(n/2) + O(1)$

It follows, $T(n) = O(n^2)$

## Solution

$$x_1 \cdot y_1 \cdot 10^n + (x_0 \cdot y_1 + x_1 \cdot y_0) \cdot 10^{n/2} + x_0 \cdot y_0$$

The goal is to decrease the number of multiplication

We can do this by observing

$$x_0 \cdot y_1 + x_1 \cdot y_0 = (x_0+x_1) \cdot (y_0+y_1) - x_0 \cdot y_0 - x_1 \cdot y_1$$

It looks that we have increased the number of multiplications, from 4 to 5.

Actually, that is not so, since we will compute $x_0 \cdot y_0$ and $x_1 \cdot y_1$ only once and then reuse it.

$T(n) = 3\, T(n/2) + O(1)$ $\boxed{T(n) = O(n^{\log 3})}$

## Problem 4

You are given an unsorted array of ALL integers in the range $[0,\ldots, 2^k-1]$ except for one integer, denoted the missing number by M.

Describe a divide-and-conquer to find the missing number *M*, and discuss its the worst-case runtime complexity in terms of $n = 2^k$.

## Solution

Partition the array wrt the most significant bit.

The total number of elements in an array is $2^k$, then in binary form one half of them starts with 0 bit and the other half starts with 1 bit.

Since one integer is missed, one partition will have an odd size. Recurs on that partition.

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

It follows, $T(n) = O(n)$.

## Problem 5

Apply the Master Theorem to $T(n) = 2T(\frac{n}{2}) + \frac{n}{\log n}$

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

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

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

The theorem does NOT apply.

The solution is $T(n) = O(n \log \log n)$.

## Solution

Consider a tree of recursive calls. At each level the work we do is given by

$$\frac{n}{\log\left(n/2^k\right)} = \frac{n}{\log n - k}$$

Since the tree height is log n, the total work in the internal nodes is

$$\sum_{k=0}^{\log n - 1} \frac{n}{\log n - k} = \sum_{j=1}^{\log n} \frac{n}{j} = n \sum_{j=1}^{\log n} \frac{1}{j} = n\, O(\log\log n)$$

Note, the work at the leaves is only O(n).



## Problem 6

Consider Strassen's algorithm for matrix multiplication. Derive and then solve a recurrence relation for the number of additions and subtractions T(n). Assume that the matrix size n is a power of 2.



## Solution

Let T(n) be the number of additions and subtractions in Strassen's algorithm. The algorithm has 18 matrix additions on each recursive call, thus

T(n) = 7 T(n/2) + 18 (n/2)^2

And its solution is

$T(n) = O(6\ n^{\log 7})$

Six times more additions than multiplications.