National University of San Agustin
System Engineer School

ADA - Lab 03 - Divide and conquer

Christian E. Portugal-Zambrano

September 16, 2018

1 Introduction

Divide an Conquer is a general algorithm design technique that solves a problem by dividing it into several smaller subproblems of the same type (ideally, of about equal size), solving each of them recursively, and then combining their solutions to get a solution to the original problem. Many efficient algorithms are based on this technique, although it can be both inapplicable and inferior to simpler algorithmic solutions [1].

2 Pre-requisites

You must have read Chapter 5 from [1] or Chapter 4 from [2], be sure to understand the substitution method, recursion three and master method to solve recurrences. The scripts for make algorithm time analysis from week 02 are already available, please make sure to elaborate plots with R (suggested) but you can use Pyplot for python.

3 Programming environment

Don't worry much about this, as your instructor I will not force you to choose a specific one, contrary to this I encourage you to select from these or another one you are already selected:

- Eclipse - https://www.eclipse.org
- Netbeans - https://netbeans.org

- Anaconda - https://www.anaconda.com
- Visual Studio (Code) - https://visualstudio.microsoft.com/es/
- Jetbrain IDE's - https://www.jetbrains.com
- Atom - https://ide.atom.io
- Sublime - https://www.sublimetext.com

All IDE's I've mentioned above are multi-platform and GUI based, but if you are a console user probably you will feel comfortable with VIM or EMACS. For this practicas as we did last week, you can use ggplot.

4 Algorithms presentation

So, as we are working with LaTeX algorithms presentation will be easy, you just need to add this at the header of your document:

```
\usepackage{algorithm}
\usepackage{algorithmicx}
\usepackage[noend]{algpseudocode}
```

Let's get start with the closest-pair problem!

4.1 Closest-pair problem

The closest-pair problem calls for finding the two closest points in a set of n points. It is the simplest of a variety of problems in computational geometry that deals with proximity of points in the plane or higher-dimensional spaces. Points in question can represent such physical objects as airplanes or post offices as well as database records, statistical samples, DNA sequences, and so on. An air-traffic controller might be interested in two closest planes as the most probable collision candidates. A regional postal service manager might need a solution to the closest- pair problem to find candidate post-office locations to be closed. One of the important applications of the closest-pair problem is cluster analysis in statistics. Based on n data points, hierarchical cluster analysis seeks to organize them in a hierarchy of clusters based on some similarity metric. For numerical data, this metric is usually the Euclidean distance; for text and other nonnumerical data, metrics such as the Hamming distance.

4.2 Warm-up:Brute-force approach

For simplicity, we consider the two-dimensional case of the closest-pair problem. We assume that the points in question are specified in a standard fashion by their (x, y) Cartesian coordinates and that the distance between two points $p_i(x_i, y_i)$ and $p_j(x_j, y_j)$ is the standard Euclidean distance

$$d(p_i, p_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

So, here we have to implemented the brute force solution to this problem following the suggested algorithm: Implement this algorithm using the IDE and language of your

Algorithm 1 Brute-Force Closest Pair algorithm: Find the distance between two closest points in a plane.

```
Require: List P of $n \geq 2$ points.
Ensure: Distance between closest points.
1: procedure BRUTEFORCECLOSESTPAIR(P)
2:     $N \leftarrow \infty$
3:     for $i \leftarrow 1$ to $i \leftarrow n-1$ do
4:         for $j \leftarrow i+1$ to $n$ do
5:             $d \leftarrow min(d, \sqrt{(}(x_i - x_j)^2 + (y_i - y_j)^2))$
6:     Return(d)
```

preference, then make an analysis of its behavior, consider test the algorithm with 100, 1000, 10000 points, make sure the points are generated randomly, elaborate the plots needed to show the results based on the time performed for each configuration.

4.2.1 To Do:Divide and Conquer for Closest Pair problem

Let P be a set of $n > 1$ points in the Cartesian plane. For the sake of simplicity, we assume that the points are distinct. We can also assume that the points are ordered in nondecreasing order of their x coordinate. (If they were not, we could sort them first by an efficient sorting algorithm such as merge-sort.) It will also be convenient to have the points sorted in a separate list in nondecreasing order of the y coordinate; we will denote such a list Q. If $2 \leq n \leq 3$, the problem can be solved by the obvious brute-force algorithm. If $n > 3$, we can divide the points into two subsets P_l and P_r of $\lceil n/2 \rceil$ and $\lfloor n/2 \rfloor$ points, respectively, by drawing a vertical line through the median m of their x coordinates so that $\lceil n/2 \rceil$ points lie to the left of or on the line itself, and $\lfloor n/2 \rfloor$ points lie to the right of or on the line. Then we can solve the closest-pair problem recursively for subsets P_l and P_r . Let d_l and d_r be the smallest distances between pairs of points in P_l and P_r, respectively, and let $d = min\{d_l, d_r\}$. Implement the algorithm 2 and do the next things:

- How much time take the algorithm with 100, 1000, 10000 and 1000000 of random points?.
- What are the plots for each test.
- What can you describe about the plots for each test?
- What would be the limits of points for a comparison between the brute-force and divide and conquer approach.
- Demonstrate the cost using master theorem and analytically.

For this practice you can use some library of code to perform the sorting needed.

Algorithm 2 EfficientClosestPair

```
Require: Array P of $n > 2$ points in nondecreasing order by x coordinate, Array Q of
    $n > 2$ points in nondecreasing order by y coordinate
Ensure: Distance between closest pair of points
procedure EfficientClosestPair(P,Q)
    if $n \leq 3$ then return BruteForceClosestPair(P)
    else
        copy $\lceil n/2 \rceil$ points of P to $P_l$
        copy $\lceil n/2 \rceil$ points of Q to $Q_l$
        copy $\lfloor n/2 \rfloor$ points of P to $P_r$
        copy $\lfloor n/2 \rfloor$ points of Q to $Q_r$
        $d_l \leftarrow$ EfficientClosestPair$(P_l, Q_l)$
        $d_r \leftarrow$ EfficientClosestPair$(P_r, Q_r)$
        $d \leftarrow \min\{d_l, d_r\}$
        $m \leftarrow P[\lceil n/2 \rceil - 1].x$
        copy all points of Q where $|x - m| < d$ en $S[0..num - 1]$
        $dminsq \leftarrow d^2$
        for $i \leftarrow 0$ to $num - 2$ do
            $k \leftarrow i + 1$
            while $k \leq num - 1$ and $(S[k].y - S[i].y)^2 < dminsq$ do
                $dminsq \leftarrow min((S[k].x - S[i].x)^2 + (S[k].y - S[i] - y)^2, dminsq)$
                $k \leftarrow k + 1$
    return (dminsq)
```

5 Deep inside

This section is just for anyone who wants to make a deep inside into the theory and like challenges[1], you will have to prepare a presentation of just 5 minutes to explain and run the algorithm, then you will have to defend yourself another five minutes of questions. I want that all students benefit from your presentation, remember that we are here to learn. If you want to do this please email to cportugalz@unsa.edu.pe

6 Deadline

For this practice one score will be taken at class time and final results presented in the next class. Remember that plagiarism must be avoided and if it is detected the grade will be zero and repetition informed to superior authorities. An unnamed pdf must be presented at the next class jointly with code and plots needed to perform the demo. All question and doubts must be done to the same email.

References

[1] A. Levitin, *Introduction To Design And Analysis Of Algorithms, 2/E.* Pearson Education India, 2008.

[2] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms.* MIT press, 2009.

[1] This must not be included into the report