Laboratory practice No. 2: Brute force or Exhaustive search

María Sofía Uribe
Universidad Eafit
Medellín, Colombia
msuribec@eafit.edu.co

Isabel Cristina Graciano
Universidad Eafit
Medellín, Colombia
icgracianv@eafit.edu.co

3) Practice for final project defense presentation

3.1 Our algorithm starts by finding all possible permutations (without repetition) given an array of elements. In this step we do not include the source because we know that said vertex will be visited at the start and at the end of the circuit, this assumption does not pose a problem because the graph is complete. We then compute the cost of each one of the circuits found and choose the least expensive one.

3.2 $\mathbf{O((V-1)!)}$ The reason this algorithm is O((V-1)!) is that instead of generating all permutations of V vertices we exclude the source vertex thus there are (V-1)! Circuits from which we must choose the cheapest one.

3.3 A graph with 50 vertices would produce 49! permutations, that is 6.0828186^{62} circuits. We don't have to compute the time it will take the algorithm to run all other operations, like compute the cost of a circuit or run other auxiliary functions, to know that the brute force algorithm is not applicable to a graph this large.

3.4 We used a list of pairs to represent the holes in the board and in order to make it more efficient we implemented a one-dimensional array instead of a matrix and instead of checking the whole board we just check the current play and then backtrack in case it is not valid.

3.5 $O(n^2)$, because we have a loop and inside of it is an if conditional with a total complexity of $O(n)$. So, we have $O(n \text{ x } n) = O(n^2)$,.

3.6 N represents the number of elements in the array.

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

4) Practice for midterms

4.1 Maximum subarray

4.1.1 *actual > máximo*
4.1.2 $O(n^2), n\ is\ the\ number\ of\ elements\ of\ the\ array$

4.2 Sorting

4.2.1 $ordenar(arr, k+1);$

4.2.2 $O(n!)\ , n\ is\ the\ number\ of\ elements\ of\ the\ array$

4.3 Search string within a string

4.3.1 *i-M*
4.3.2 *N*
4.3.3 It takes $O(n+m)$ $but\ because\ M \geq N\ we\ could\ say\ the\ complexity\ in\ Big\ Oh\ Notation\ is\ O(n)$

4.4

4.4.1 Temp%10
4.4.2 *b. O (|N−M|).log10 M*

4.5

4.5.1 $i+1$

4.5.2 $left == right$

4.5.3 $O(n^2)\ , wheren\ n\ is\ the\ number\ of\ elements\ in\ the\ array$

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

5) Recommended reading (optional)

5.1 Summary

A minimum spanning tree (MST) consists of the minimum number of edges necessary to connect all a graph's vertices.

On the other hand, brute force is a straightforward approach to solving a problem, this is one of the easiest algorithms to solve a problem (Just in case that you can use it), the bad news about it is that the time is exponential.

Selection sort: One of the most famous algorithms is Selection Sort, for example, if we want to organize numbers from the smallest to the largest we start by scanning a given list of integers to find the smallest number and if that number is smaller than the one that is in the first place we can exchange them. Then, do the same process with the second one and the rest of them. Thus, selection sort is O(n^2) in all cases.

Closest-pair problem: given a set of n point(which in real life can be airplanes , post offices etc.), find a pair of points with the smallest distance between them. To do this we calculate the distance between each pair of distinct points (P1,P2) with the standard Euclidean distance.
Convex-Hull problem: Given a set of n points we need to construct the convex hull. To solve it we need to find the points that will serve as the vertices of the polygon in question, then we will create a line segment connecting two points and then those two will be part of the convex-hull. Now, if we follow the same path connecting the rest of the vertices and we will get a complete convex-hull.

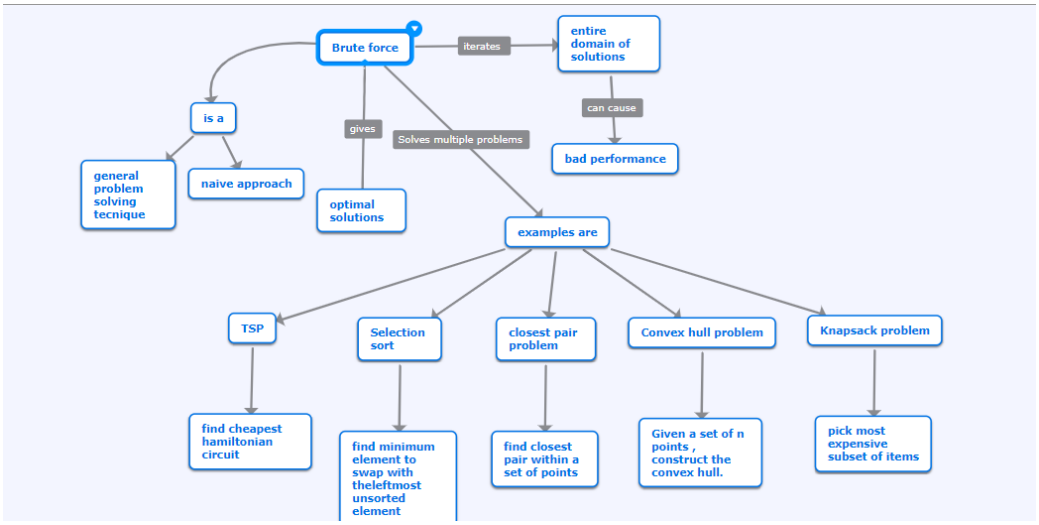
Exhaustive Search: this approach to combinatorial problems so we generate each element of the problem and select one by one those who satisfy the constraint.

Travelling salesman problem: Is defined as a Hamiltonian circuit so we need to find the cheapest way to visit every node beginning and ending in the same node.

Knapsack problem: Given n items of known weights, values and a knapsack with W capacity we need to find the most subset of the items that fit int the knapsack. To solve it we implement exhaustive-search approach generating all the possible answers and choosing the best to solve the problem.

Concept diagram based on the key theoretical elements of "Robert Lafore, Data Structures and Algorithms in Java (2nd edition), Chapter 13: Graphs. 2002"

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473



6) Team work and gradual progress (optional)

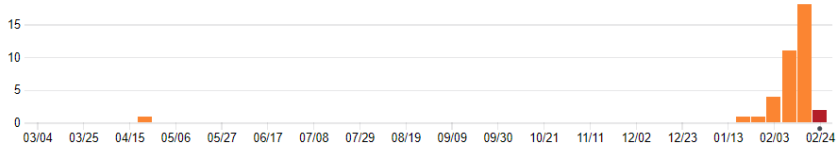
6.1 Meeting minutes

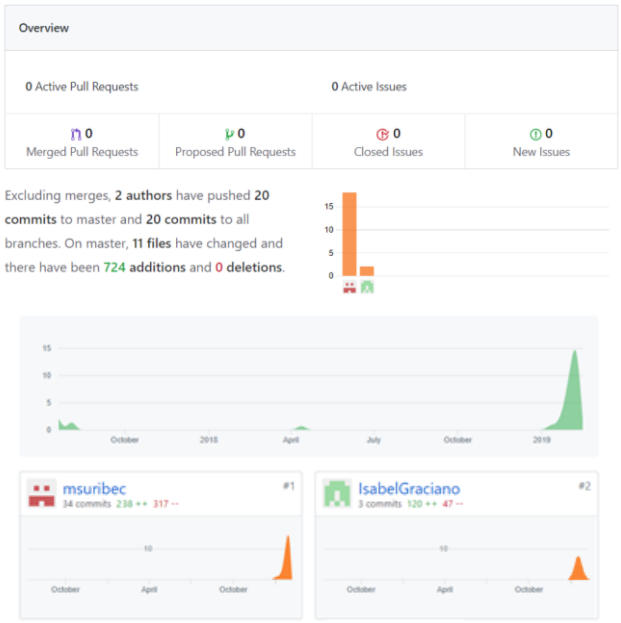
TEAM MEMBER	DATE	DONE	DOING	TO DO
ISABEL	21/02/2019	Discussing task distribution		Plan data structure for exercise 1,2 in the workshop
SOFIA	21/02/2019	Discussing task distribution		Plan data structure for exercise 1,2 in the workshop
ISABEL	23/02/2019	Data structure works	Solve online exercise	Optional Reading (summary)
SOFIA	23/02/2019	Data structure works	Solve online exercise	Optional Reading(concept map)
ISABEL	24/02/2019	Summary of reading		
SOFIA	24/02/2019	Concept map		
ISABEL	24/02/2019	Answer exercise 3.1-3.3		Code comments
SOFIA	24/02/2019	Answer exercise 3 .4-3.6		Upload code
ISABEL	24/02/2019	Code comments		
SOFIA	24/02/2019	Upload code and report		

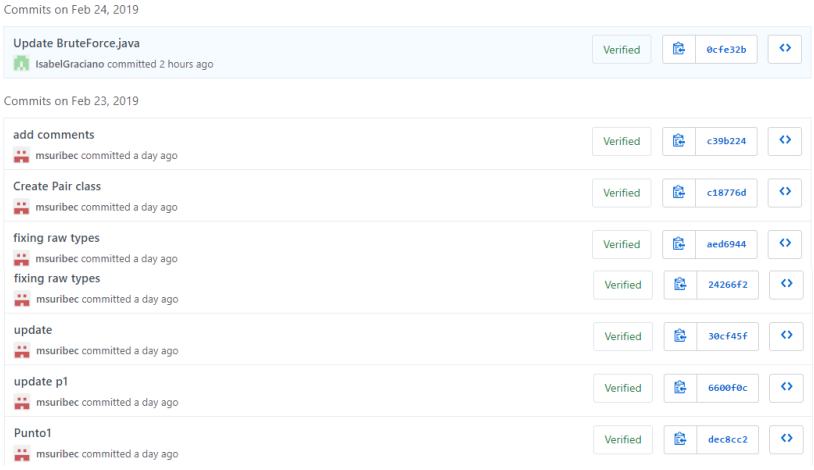
6.2 History of changes of the code

PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473

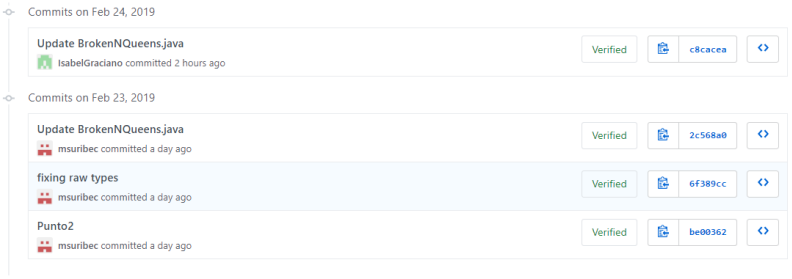
ESTRUCTURA DE DATOS 2
Código ST0247



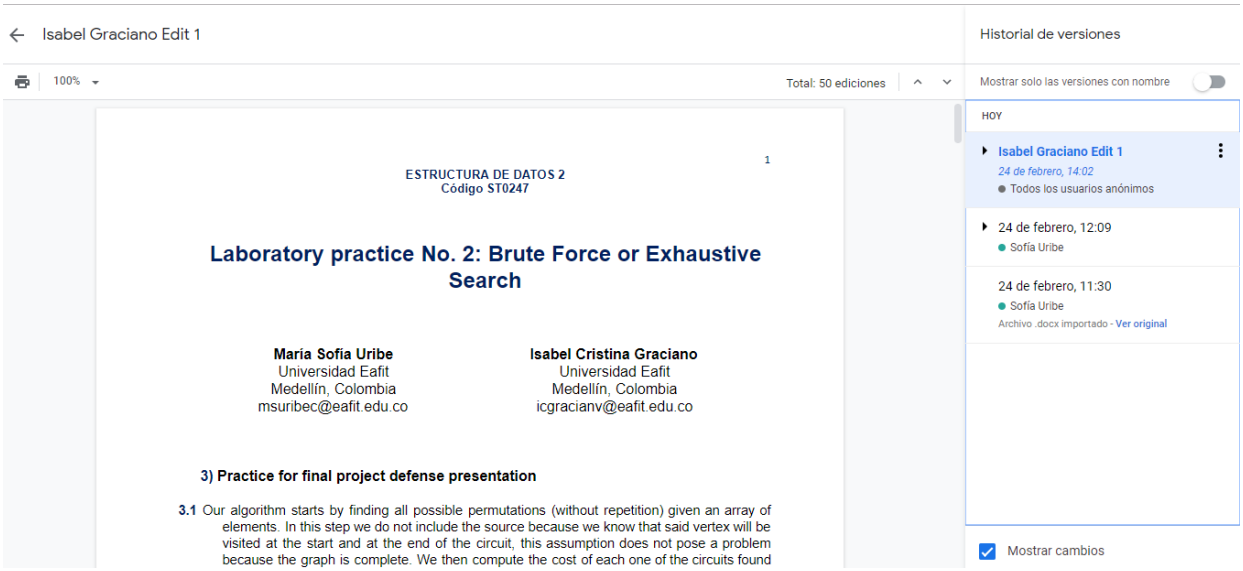




PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473



6.3 History of changes of the report



PhD. Mauricio Toro Bermúdez
Professor | School of Engineering | Informatics and Systems
Email: mtorobe@eafit.edu.co | Office: Building 19 – 627
Phone: (+57) (4) 261 95 00 Ext. 9473