ALGORITHMS AS A SOLUTION TO TRAFFIC

David Jose Cardona Nieves
Universidad Eafit
Colombia
djcardonan@eafit.edu.co

Juan David Valencia Torres
Universidad Eafit
Colombia
jdvalencit@eafit.edu.co

Mauricio Toro
Universidad Eafit
Colombia
mtorobe@eafit.edu.co

ABSTRACT

The enigma to be solved in the realization of this project is to find a way to reduce the emission of greenhouse gases, a very worrying issue nowadays. In order to solve this problem, the use of algorithms in the realization of efficient routes both at a productive and ecological level is proposed. However, this project has a great limitation (and mainly, a great challenge) and this is not being able to establish the routes in an almost apotheosis way due to the mobility restriction of the vehicles because of their need to recharge the battery. The expected result is to be able to have an efficient algorithm that can be compared with others performed in alternative exercises such as eVRP.

Keywords
Graph, Route optimization, mapping, eVRP, nearest neighbor.

CM classification keywords

Theory of computation → Design and analysis of algorithms → Graph algorithms analysis → Shortest paths

1. INTRODUCTION

We cannot deny that the ecological field has been gaining power in our will, do and think with the passing of time. This is how ideas for a better care of the environment have been forged in the minds of millions of people.

Every day we can see how new technologies such as electric vehicles are born and grow. However, these technologies need to be regulated and improved over time. For example, we can talk about electric vehicles and how we could contribute a lot to the environment if we make this technology efficient.

2. PROBLEM

We must anticipate the future by seeking to solve the great question of this century: How to maintain the balance between life and development?

More in detail, we can talk about how to accurately and efficiently establish the routes to be used by cargo vehicles, transport, etc... To reach a destination by traveling through each of the places we tell it to travel. Bringing this to reality, we would talk about how a hypothetical autonomous vehicle could pick up staff of a company in their respective addresses to take them to work, for example.

3. RELATED WORK

3.1 **Solving the capacity-constrained vehicle routing problem using a two-stage metaheuristic procedure.**

This document proposes a solution to the vehicle routing problem by applying what they know as "flow limits", dividing it into two phases: the first one where the route design process is carried out and the second phase where fleet planning is performed.

3.2 **A Vehicle Routing Problem with a Time Windows Approach to Improve the Delivery Process**

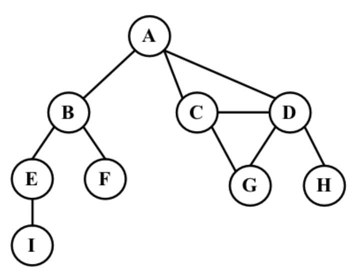
Pineda and Carabalí seek to solve the VRP (Proposed Exercise). They do this based on the Clarke-Wright Stochastic Heuristic algorithm. This algorithm has to have in its majority of implementation cases a complexity of O(n^3).

3.3 **Vehicle routing model for the distribution of Veterland Laboratories, Callbest Laboratories and Marlioü Paris Cosmetics.**

The topic addressed by Andres Felipe in his article consists of proposing a routing model for product distribution, proposing it in two phases: an initial one in which the map is sketched and a second one to generate all the routes. The algorithm used to solve this problem was the Tabu algorithm, which proposes a possible solution and then looks for a better one, if applicable. The results obtained prioritize a shorter time in each route.

4. MATRIXES AS EFFICIENT STRUCTURES

4.1 Data structures



	A	B	C	D	E	F	G	H	I
A	0	1	1	1	0	0	0	0	0
B	1	0	0	0	1	1	0	0	0
C	1	0	0	1	0	0	1	0	0
D	1	0	1	0	0	0	1	1	0
E	0	1	0	0	0	0	0	0	1
F	0	1	0	0	0	0	0	0	0
G	0	0	1	1	0	0	0	0	0
H	0	0	0	1	0	0	0	0	0
I	0	0	0	0	1	0	0	0	0

The main data structure on which this solution will be based is a multidimensional adjacency matrix where different variables will be stored to connect each of the points on the map and to be able to operate with these connections. All this in order to achieve a really efficient data access capacity.

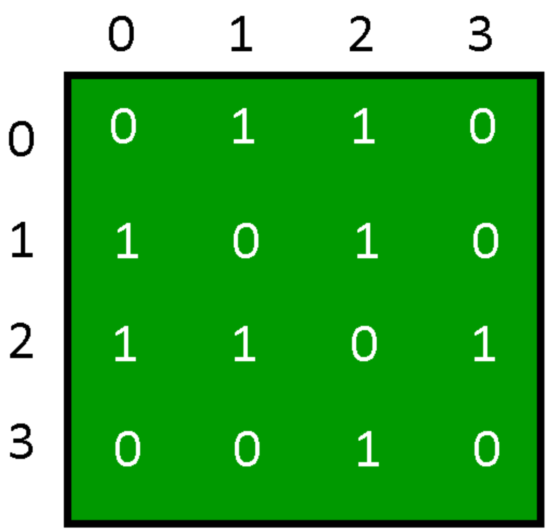


Figure 1: Connection matrix between different points (1: there is connection, 0: there is no connection) (Replaced in future development by the distance)

4.2 Operations of the data structure

The operations to be performed within the data structure are:

- Search
- Addition of nodes
- Node deletion

4.3 Design criteria of the data structure

We chose this data structure because of its ease of searching and its very efficient algorithmic complexities. That is, we ended up gaining a lot of time efficiency at the cost of sacrificing a few extra megabytes in memory (all this taking into account that the solution may not be the most efficient when talking about very large amounts of data).

4.4 Complexity Analysis

Node search	O(1)
Search for nearby neighbors	O(n)
Graph creation	O(n^2)
Total	O(n^3)

where n is the number of nodes.

4.5 Nearest Neighbor Algorithm

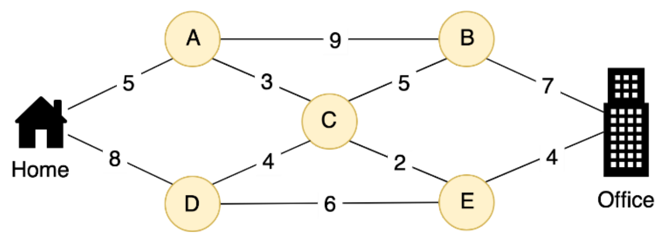


Figure 3: An illustration of the "blueprint" of the NNA algorithm in route development (and optimization).

4.6 Algorithm complexity calculation

Calculate the complexity of the algorithm for the worst case, the best case and the average case

Sub-problem	Complexity
Matrix creation	O(n^2)
Search for nearby neighbors	O(n)
Creation of roads	O(n^3)
Total Complexity	O(n^3)

Note: The complexity is O(n^3) by sum rule.

Table 2: Table of complexities (separate and total) where n represents the number of nodes.

4.7 Algorithm design criteria

The algorithm was designed in this way to seek to "humanize" the optimization criteria and make it possible to obtain good results even when running on not very powerful machines. This is why the algorithm is based on a greedy solution that seeks to propose optimal solutions formed from equally optimal "smaller" solutions. This solution is then joined together with the others to, in theory, find the best possible or at least a functional one.

4.8 Execution Time

	Dataset 1	Dataset 2	Dataset 3
Best case	400 ms	1800 ms	1820 ms
Average case	406 ms	1950 ms	1960 ms
Worst case	415 ms	2100 ms	2100 ms

Table 3: Algorithm execution times with different data sets.

4.9 Memory consumption

	Dataset 1	Dataset 2	Dataset 3
Memory consumption	12MB	14MB	13MB

Table 4: Algorithm memory consumption with different datasets

4.10 Result analysis

Time	Memory	Nodes	Vehicles
11s	21MB	2	1
13s	28MB	345	30
15s	29MB	359	26

Table 5: Analysis of the results obtained with the implementation of the algorithm.

5. NNA for eVRP

The data structure and algorithm are explained below.

5.1 Data structure

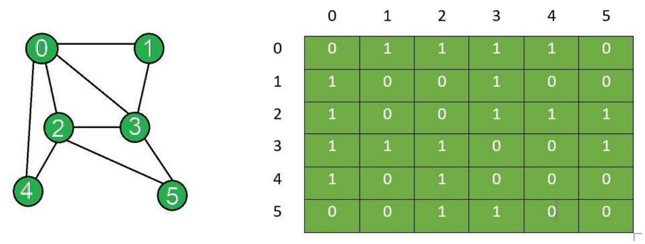


Figure 4: Adjacency matrix and the graph that represents it

5.2 Operations of the data structure

	NODO 1	NODO 2	NODO 3
NODO 1	-1	-1	-1
NODO 2	-1	-1	-1
NODO 2	-1	-1	-1

	NODO 1	NODO 2	NODO 3
NODO 1	1	1	1
NODO 2	1	1	0
NODO 3	1	0	1

Figure 5: Matrix creation operation

5.3 Design criteria of the data structure

We decided to use an adjacency matrix as the data structure of the project since it helps us to improve the efficiency of the project, which we aim to impact the final performance of the project. Things like less complexity in accessing the weight of an edge help greatly in employing the nearest neighbor method and thus conforming to the expected parameters. A great advantage of using this type of matrix in this case is to be able to save both time and memory since if we were to use, for example, adjacency lists, it would be even tedious to operate with certain nodes since "theoretically" they would not be connected (which the exercise from the beginning tells us is false). Moreover, it is the structure used in most (if not all) of the developments cited in the bibliography consulted for this project.

5.4 Complexity analysis

	Complejidad
Node search	O(1)
Search for nearby neighbors	O(n)
Graph creation	O(n^2)
Add path between nodes	O(1)

Table 6: Complexity reporting table

5.5 Algorithm

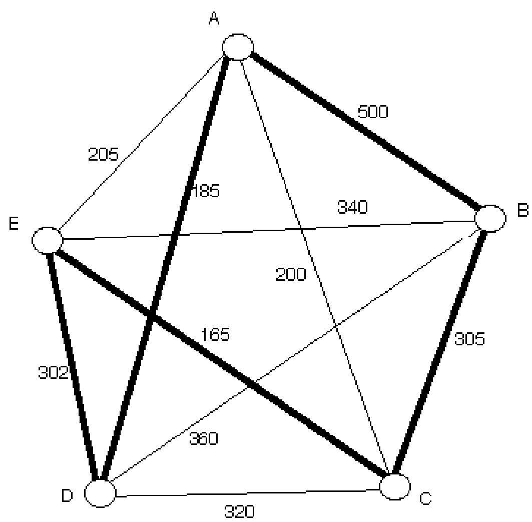


Figure 6: Step-by-step assembly of DNA fragments using Bruijn's graphs.

5.6 Complexity analysis of the algorithm

Sub-problem	**Complexity**
Create the Bruijn network with the following sequences	O(N)
Updating the Bruijn network with the sequences	$O(A.N^2)$
Finding the genes	O(V)
Total Complexity	$O(A.N^2 + V)$

Table 7: Complexity of each of the subproblems composing the algorithm. Let A be the length of a DNA sequence, N the number of DNA sequences, and V the number of different K-numbers obtained from the DNA sequences.

5.7 Design criteria of the algorithm

The algorithm chosen to solve the routing problem was the NNA (Nearest Neighbor Algorithm) because it provides fairly optimal solutions (without assuring that they will be the best), greatly reducing the high processing costs required by other solutions, such as genetic algorithms or brute force. In addition, by performing this exercise using this type of algorithms we found a great amount of information that allowed us to improve our development little by little, even relying on "greedy" solutions provided by other algorithms for minor problems we encountered (such as data import, data reading and node sorting).

.

5.8 Execution times

	Dataset 1	**Dataset 2**	**Dataset 3**
Best case	11s	15	16s
Average case	14s	17	22s
Worst case	16s	22s	31s

Table 8: Algorithm execution times with different data sets.

5.9 Memory consumption

	Dataset 1	**Dataset 2**	**Dataset 3**
Memory consumption (MB)	12	14	13

Table 9: Algorithm memory consumption with different datasets.

5.10 Analysis of the results

	Best	Average	Worst
Dataset 1	**11s / 12MB**	**14s / 12MB**	**16s / 12MB**
Dataset 2	**15s / 14MB**	**17s / 14MB**	**22s / 14MB**
Dataset 3	**16s / 13MB**	**22s / 13MB**	**31s / 13MB**

Table 10: Analysis of the results obtained with the implementation of the algorithm

6. CONCLUSIONS

Concisely, with this project we sought to focus on the development of an algorithm to solve an environmental problem through the creation and optimization of routes for deliveries. (applicable to multiple situations). It was an arduous process to develop this digital solution because the available information, although extensive, was often confusing and ended up proposing the use of other methods.

Sin embargo logramos resultados muy buenos sin usar cantidades de recursos extremas, al punto de lograr manejar los datasets dados en cuestión de segundos. Finalmente, podemos decir que la solución que planteamos inicialmente difiere totalmente de la solución final, siendo que nuestra However, we achieved very good results without using extreme amounts of resources, to the point of managing the given datasets in a matter of seconds. Finally, we can say that the solution we initially proposed differs completely from the final solution, since our first option was the development of an adaptation of Dijkstra's algorithm.

6.1 Future work

With this project we accomplished what we set out to do in the project done last semester, so, hoping to continue like this, we would like to improve our "formal part" of the project (reports, comments, etc) and learn more about dynamic programming to be able to apply it in this exercise and those to come in the future.

REFERENCES

1. Velandia, J. Felipe. 2018. Propuesta de un modelo de ruteo VRP para una empresa OPL de medicamentos y suministros médicos en Bogotá.

2. Daza, J. Montoya, J. and Narducci, 2009. RESOLUCIÓN DEL PROBLEMA DE ENRUTAMIENTO DE VEHÍCULOS CON LIMITACIONES DE CAPACIDAD UTILIZANDO UN PROCEDIMIENTO METAHEURÍSTICO DE DOS FASES. 12th ed. Medellín: Escuela de Ingeniería de Antioquia, pp. 23-34.

3. Carabalí, H. and Pineda, U. 2020. Un Problema de Enrutamiento del Vehículo con Enfoque de Ventanas de Tiempo para Mejorar el Proceso de Entregas. Ingeniería