Reducing Traffic and Pollution Utilizing Carpooling

Rafael Gomez
Universidad EAFIT
Colombia
rgomeze@eafit.edu.co

Stiven Agudelo Osorno
Universidad EAFIT
Country
sagudeloo@eafit.edu.co

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

ABSTRACT

We were tasked to develop an algorithm to find an efficient way to carpool within a university or company in order to reduce the number of cars on the road and therefore the traffic in the city. The problem is important because if we can find a way to carpool and the people don't have to take much longer to get to work then they are much more likely to carpool. It is also important because if we can effectively carpool, we can reduce traffic which has been shown to reduce the quality of life, quality of the air we breathe and even reduce the productivity of people working. Examples of similar problems are transporting food to their destinations, routes for doctors that perform their service at the residency of the patient, developing routes to deliver mail and finally the most applicable problem is how Uberpool can determine their routes for their application. After considering how the problem was proposed we decided to change the value in which the person can decide how much longer their route will take if they decide to carpool. What this means if we were to use an example of an application, someone could choose when deciding to carpool that if it takes less than 15 minutes in addition to the original time, they would take to get to their destination then they will carpool. With the problem that was presented with 205 employees, which means the original number of cars is 205 and the point of the algorithm is to reduce the number of cars so we'll consider the result as the number of cars needed to get the 205 people to their job, the results were as followed with (p = limit of extra minutes to carpool); p = 5 (44 cars), p = 10 (42 cars) and p = 13 (41 cars; which is the lowest number of cars possible), interestingly enough if we apply p = 0 which means nobody would take any longer to get to their location; the result is 111 cars which is almost half of the original 205 cars. We can conclude that carpooling could be an effective way of reducing the number of cars on the road and therefore the traffic.

1. INTRODUCTION

Although carpooling has existed for quite some time, it has never really been applied in a large scale. Now with more cars than ever on the road, we have two very realistic problems which are the following; first and foremost, with the amount of cars increasing we are now seeing much higher levels of traffic and therefore increased commute times, one solution can be organized carpooling in each job.

2. PROBLEM

We were tasked to develop an algorithm that a company could use to come up with a carpooling route between all the car owners of said company. The algorithm wants to know what is the fewest number of cars needed to get everyone to work with the maximum number of occupants being four in any given car. We are to assume that the driver will always take the shortest route to work and only go to work for that day.

3. RELATED WORK

3.1 Getting You Faster to Work – A Genetic Algorithm Approach to the Traffic Assignment Problem

The last decades saw a constant increase in road traffic demand. Trips tend to concentrate on a few central spots, like major roads or large intersections during the rush hour periods. This is very likely to cause congestion in the road network. This paper has demonstrated that genetic algorithms area feasible approach to the route assignment problem in road networks. This optimization is performed with a global view of a microscopic traffic model.[1]

3.2 Order Assignment and Routing for Online Food Delivery: Two Meta-Heuristic Methods

There is a critical need to improve service level (in terms of delivery time) with less delivery cost. For this end, the key problems include the assignment of the orders to the delivery staff and routing of the orders for each staff. The hierarchical method reduces the total traveling distance by at least 11 percent for the instances, which implies a reduced customer waiting time and delivery cost.[4]

3.3 Multi-Objective Optimization for Reducing Delays and Congestion in Air Traffic Management

Nowadays, with the increasing traffic density of the European airspace, air traffic management includes the planning and monitoring way to facilitate the work of the air traffic controllers. This article presents the use of an evolutionary multi-objective algorithm for optimizing a schedule on the time of overflight of the way-points. An evaluation function takes around 3 seconds with the characteristic function method. At this point of the research, we know that the mean fitness of the population increases along the generations.[2]

3.4 An Algorithmic Approach for Environmentally-Friendly Traffic Control in Smart Cities

With the constant development and urbanization of human society in recent years, the density of urban

residential areas is increasing all over the world. To serve these increasingly high-density urban populations, governments everywhere seek sustainable development approaches [13] that balance commercial and personal needs with environmental protection. Discussed that a fully central optimum solution is likely to be NP-hard and showed our fully distributed algorithm has a bounded performance.[3]

4. Carpool Algorithm with Dictionary

4.1 Data Structure

The first data structure that we used was a dictionary that had the all the people as keys in the dictionary and the values were a list that was composed of tuples that in the first position had the destination or the next person to be visitied and in the second position had how many minutes it would take to get to that person.

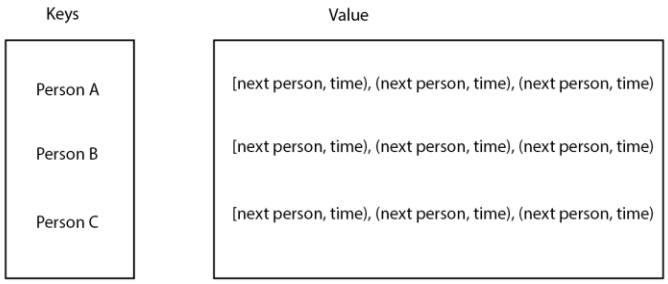

Figure 1: Python dictionary that has keys set to all the people and the values are the tuples with the next person and the time to get there.

4.2 Operations of the Data Structure

How the operations worked, was it looked at all of the nodes and found the closest person, after that it would look at the closest person to the last person that entered the car, all while calculating that if we add the person to the route whether or not the time would be under the total allowed for the person with their original time to the destination.

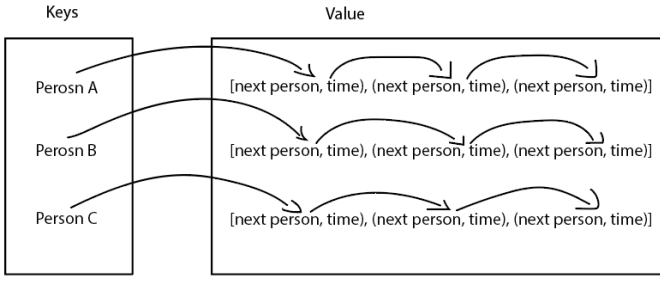

Figure 2: Find other people that the person can carpool with.

4.3 Design Criteria of the Data Structure

The main reason why I chose to use a dictionary at first was that in order to find or return a value in the dictionary is fast O(logn) and the code in order to do the operations and find if someone could be added to the car could be done easily.

4.4 Analyzing the Complexity

When analyzing the complexity, we realized that with this first data structure and algorithm that we needed three loops to find and calculate if the person could be added to the car. That gave us a time complexity of O(n^3) which was far too great.

4.7 Analyzing the Results

The results we had with this algorithm left much to be desired. For the situation that there were 205 people the program took almost 30 seconds to execute and for p = 1.2 only lowered to 74 cars which was very inefficient and was when we decided to change the data structure and algorithm.

5. Carpooling Algorithm

5.1 Data Structure

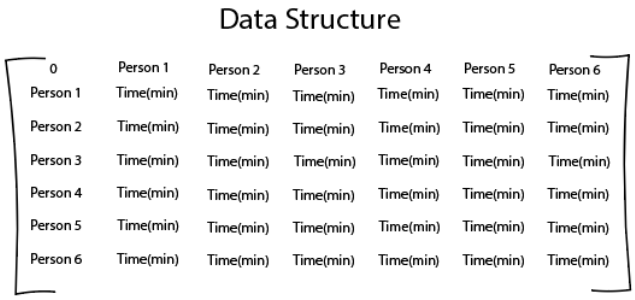

Figure 3: Matrix with the first row and column as the people and inside the matrix is how much time to go from the person in the row to the person in the column.

5.2 Operations of the Data Structure

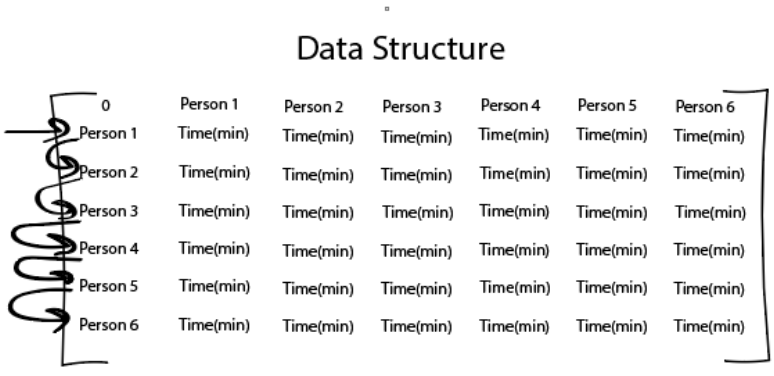

Figure 4: Find the person farthest from the destination, used for the first person of the car (to begin the car).

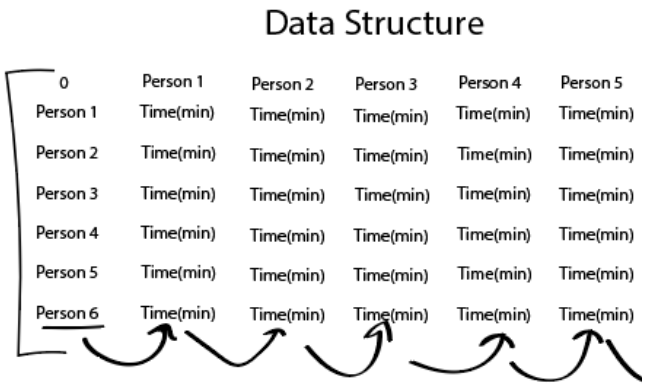

Figure 5: Find the closest destination (column) from the person (row), used to find the next passenger.

5.3 Design Criteria of the Data Structure

The data structure that we decided to use was a matrix. We decided to use a matrix because in an example such as this that we have the information to have a completely filled mapped; in other words, we have the distance in time from every person to every other person. The only zeros in the matrix is the primary diagonal because the distance from the person to themselves is zero; this means that a matrix is very efficient. Another variable in choosing a matrix is that it supplies all the information that we need with minimal need of variables. In the figures it shows the first row and column as person; however, it is just to give a visual representation. Every value is distance in time, and we know who the person is by the number of the column or row. This means that the matrix is filled with just 8-bit integers helping to reduce memory consumption.

5.4 Analyzing the Complexity

Method	Complexity
fill_all_cars	O(n)
fill_cars	O(n^2)
find_farthest_person	O(n)
find_closest_person	O(n)
find_any_possible	O(n)

Table 1: Table analyzing the complexity.

5.5 Execution time

Method	U = 11	U = 205
fill_all_cars	0 ms	65 ms
fill_cars	0 ms	65 ms
find_farthest_person	0 ms	0 ms
find_closest_person	0 ms	0 ms
find_any_possible	0 ms	0 ms
file_reader	0 ms	50 ms

Table 6: Execution time of each method in the algorithm for each data set.

5.6 Memory used

	U = 11	U = 205
Memory Usage	1.9 kB	656 kB

Table 7: Memory used for each method in the algorithm for each data set.

6. CONCLUSIONS

To summarize this report, we will go over the problem and the process we used to develop an algorithm for a desirable result. The original problem we were tasked to resolve was; there is too much traffic in the city, so in order to reduce traffic, we the employees of a company are going to begin carpooling; however, when people decide whether or not to carpool a big deciding factor is of course the question, how much longer will it take me to get to work if I have to pick up another person or more than one person. To resolve this, we implemented an algorithm that the user can enter in how much longer (in minutes) they are willing to take to get to work.

In order to solve this problem, we were asked to take the people and organize them into cars, a maximum of five people per car. When you look at the results when considering how much longer the person is willing to take to get to work its rather interesting. Consider the value p as

the value in minutes how much longer the person is willing to take to get to work; p = 5 means that the person is willing to take up to five minutes longer to get to work to be part of the carpooling program at the job. The results; if p = 13 minutes we can reduce the number of cars from 205 to 41 cars which is the absolute minimum number of cars needed to transport said 205 people to work; if p = 5 minutes we go from 205 cars to 44 cars; and an interesting result that we discovered was that if we use p = 0, meaning that the person is not willing to take any more time than they would going to work by themselves, we can still reduce the amount of cars by almost half, from 205 to 111 cars. With these results is shows that if companies and schools throughout the city decided to use carpooling that we could drastically reduce the number of cars and therefore reduce traffic, reduce commute times, improve air and life quality.

Our first solution to the problem had many issues; the main one being that the complexity in order to do all of the methods would be at least O(n^3) which means that when we were using the dataset with 205 employees it was taking longer than 20 seconds to execute the program. It was at that time that we had to reevaluate how to resolve the issue at hand; it was then that we realized that because we have a graph that is completely filled, meaning we have the distance from each person to every other person. This means that a matrix would be the most efficient data structure because accessing the info in a matrix is O(logn), normally a matrix is not efficient because a there will be a lot of zeros throughout the matrix.

In order to improve this algorithm we would make it applicable to more people than just people within the same company, meaning more user input based, meaning to input their current location and destination and then have other users input their location to find nearby people with whom to carpool with. This would be similar to Uberpool, meaning that you could do it with people outside of just a small circle making it more widespread and therefore having a larger impact.

6.1 Future Steps

The next step I would take with this program is to allow the user to input the information. So as a rider requests to carpool it can input the location and look at the other users in the area to calculate routes with all the people considered.

REFERENCES

1. Daniel Cagara, Ana L.C. Bazzan, Björn Scheuermann Getting you faster to work: agenetic algorithm approach to the traffic assignment problem in Proceeding GECCO Comp '14 Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation (Vancouver, BC, Canada — July 12 - 16, 2014) Publisher: ACM, New York, NY, USA 2014, Pages 105-106.

2. Gaétan Marceau, Pierre Savéant, Marc Schoenauer Multiobjective Optimization for Reducing Delays and Congestion in Air Traffic Management in Proceeding GECCO '13 Companion Proceedings of the 15th annual conference companion on Genetic and evolutionary computation (Amsterdam, The Netherlands—July06-10, 2013) Publisher: ACM New York, NY, USA Pages 187-188

3. Keyvan Moghadam, KaiHuang, Bhaskar Krishnamachari An Algorithmic Approach for Environmentally-Friendly Traffic Control in Smart Cities in Proceeding Urban GIS'15 Proceedings of the 1st International ACM SIGSPATIAL Workshop on Smart Cities and Urban Analytics, (Bellevue, WA, USA— November 03 - 06, 2015) Pages 14-27