Evolutionary algorithm for traffic light synchronization in "Corredor Garzon"

Alvaro Acuña, Efrain Arreche

Universidad de la Republica , Uruguay

Abstract

This Article presents the study of a traffic light synchronization optimization as a multi-objective problem, the design and implementation of an evolutionary algorithm to resolve it with high numerical efficiency and computational performance. It uses the "Corredor de Garzon" (Montevideo, Uruguay) that has a large amount of junctions, streets and traffic lights, making it an interesting and real scenario to choose from, not only from the point of view of the complexity but because the authorities responsible for the "Corredor de Garzon" had admitted that the objective (that was mainly to speed up the area) has not been accomplished.

The experimental analysis compares the solutions found using the proposed algorithm versus the data gathered in-situ. The results show that the evolutionary algorithm is able to efficiently reach significant improvements in the total cost, outperforming the reality in up to 21%.

Keywords: Evolutionary Algorithm, traffic light synchronization, real scenario, Cluster

1 Introduction

1.1 Motivation and context

Evolutionary Algorithm has shown his usefulness in complex problems and in particular the traffic lights synchronization problem. This work will develop an algorithm to resolve it with good metrics.

We chose Corredor Garzon because it's a real complex scenario with 6.5km length, 19 intersections, exclusive corridor and traffic light rules for busses, etc. And one important thing for us to strongly consider doing this project is that it has been said multiples times [20, 21] that the main objective, which was to improve the speed of public transportation has not been reached and it had problems with the traffic lights synchronization. And even 3 years later of its inauguration, it hasn't been done so any improvement could potentially help everyone.

1.2 Objectives

These are the main objectives of our work:

- Study the traffic lights problem
- Research related work
- Create an evolutionary algorithm that resolve the problem
- Create an accurate map and related configuration (traffic lights, traffic density, bus schedules) to represent the reality of the problem
- Use high performance computer techniques to speed up the process

2 Theory

2.1 Evolutionary Algorithms

Evolutionary algorithms are non-deterministic methods inspired in the natural evolution using concepts as population, crossover and mutation. This are used to resolve optimization and search problems among others.

It's an iterative technique that aims to improve the solution in each step. It's a really good technique to use in complex problems

2.2 Genetic Algorithms

The Genetic Algorithm is one of the most popular evolutionary algorithms. The base idea is that starting from a certain population, it selects the best ones based in some metric and those are used to generate new ones by combination or modification. In each step it finds better solutions until it's stopped by some criteria like number of iterations.

An individual is the candidate solution for the optimization problem, and the fitness function determines the quality of that solution. The starting population of individuals could be generated randomly or by many other ways (i.e. in our case is made by random modifications of the actual solution). In each generation it applies operators like crossover or mutation to modify the individuals, then it selects the best ones and so on.

This is not an exact technique but it reaches really good approximations and it's suited for complex problems because of its flexibility and robustness.

2.3 Parallel Genetic Algorithms

Complex problems require a high computer power so the parallelization techniques are useful to reach good execution times. In the genetic algorithm plenty of time is invested in the fitness function so it's a common practice to try to distribute the load in different processors because it will lead to a very good improvement.

We used the Master-Slave technique, the master is in charge of distributing the load and the slaves are the ones that execute the fitness function.

2.4 Vehicular Transit Problem

There is a global tendency in the number of cars growth that will lead to problems that affect the quality of life related to the vehicular traffics congestion. [2]
So it's important to search methods which aim to improve the average speed in the street and try to stop the development of traffic jams and congestion

2.5 Corredor Garzon

Corredor Garzon is located at North-East of Montevideo, Uruguay. It was built as part of a mobility traffic project that included four other corridors within the city. It connects from beginning to end Paso Molino and Colon two very crowded neighborhoods. With a 6.5km length it goes through very important streets such as Millan (which goes directly to the highway 5) and many others.

The corridor consists basically in three parallel and independent streets; two of them are one way and have two lanes each and the other one goes in the middle of the other two and is two-way with only one narrow lane each way **and is exclusive for busses**.

Near the corridor there are no clear parallel streets so it's possible to turn left on some streets of

the corridor, even for the central lane, this is not a common thing in a corridor and it's a big problem trying to speed it up this way.

2.6 Simulation tools

Traffic simulators are widely used to research traffic congestion or measurements about the impact of new infrastructures. There are clear advantages on using simulations; they are cheap, fast and flexible.

SUMO [15] is microscopic traffic simulator, this means that it simulate each vehicle in particular. It uses xml files to configure the map, routes, trips, traffic lights, etc.

2.7 Field Work

We did the gathering of data in the Corridor to research the data traffic. It covers 5 intersections: Camino Ariel, Battle y Ordonez, Plaza Videla, Camino Duran and Aparicio Saravia. We choose a sunny weekday in the afternoon as is the most common traffic situation (not peak hour).

Also to get the traffic light configuration we went to the corridor and get the data in-situ. Then we validate the simulation speed data with real trips over the corridor and the average bus speed was validated using GPS bus information provided by Intendencia Municipal de Montevideo [5].

3 Related Work

3.1 Introduction

The related work research goals were to study the different solution to resolve the problem and to found new practices, algorithms, and tools to help our current solution.

The traffic light optimization problem can be resolved in many ways like neural nets [6], fuzzy logic [4], Petri nets [1], etc; so the amount of solutions found was huge and diverse, for this reason we focus our research in genetic algorithms that we consider were interesting. Here are some of the works that we found:

- J. Sanchez, M. Galan and E.Rubio. Genetic algorithms and cellular automata: A new architecture for traffic light cycles optimization.2004. This work it's based in three points, first it uses genetic algorithm to optimize, cellular automaton to simulate the traffic and a cluster to execute in parallel. It's a small model with 5 street and 2 intersections. It compares the results with a random and fixed simulation founding that the solution has better results.

- J. Sanchez, M. Galan and E.Rubio. Applying a traffic lights evolutionary optimization technique to a real case: "Las Ramblas" area in Santa Cruz de Tenerife. 2008. This work apply genetic algorithm to a real case to validate the results. It has some improvements and also it's a lot more complex that the previous work. It has 42 traffic lights, 26 input streets and 20 output streets. The local government gave the data to simulate and it was used to compare against the solution, it got an improvement of 26%

- J. Penner, R.Hoar and C. Jacob. Swarm- based traffic simulation with evolutionary light adaptation. 2002. This work uses a simulation model based on swarms that are optimized using a genetic algorithm. It evaluates a complex scenario of 28 traffic lights and 9 intersections getting an improvement of 26%.

- A. Vogel, C. Goerick and W. von Seelen. Evolutionary algorithms for optimizing traffic signal operation. 2000. The solution uses a self-adaptable algorithm to improve the traffic in short and long term. It shows that even when we found the best traffic light configuration it's almost impossible that it will be useful in other situation or extreme cases like pick-hours. To solve this they propose a self-adaptable algorithm that takes real time data using sensors to detect vehicles.

They conclude that the new solution has a good performance and results.

4 Solution

4.1 Architecture

The next diagram shows the architecture used in our solution.

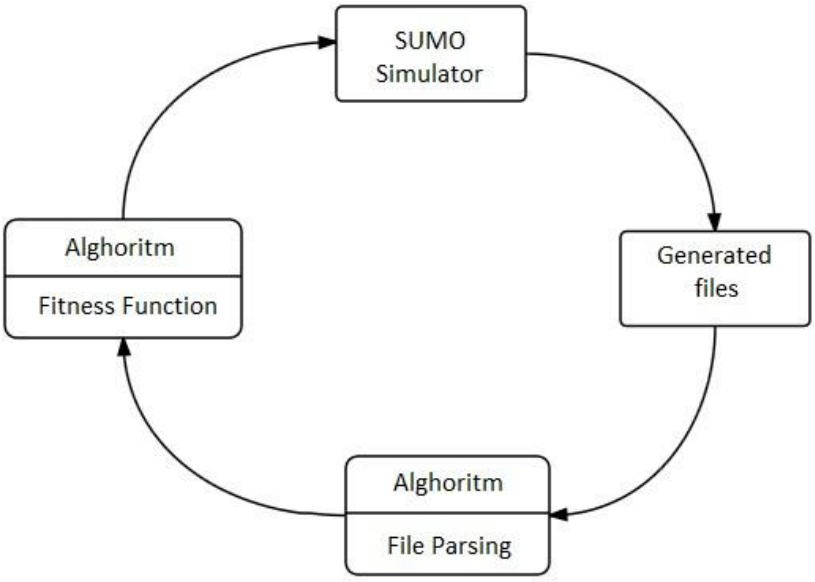

Figure 1: Architecture

Malva Library [16] it's used to implement the algorithm, and in each fitness evaluation it calls the SUMO simulator. The simulator generates files with data about the simulation, then it's processes to get useful information like average speed that it's going to be used in the fitness value.

4.2 Genetic algorithm implemented.

Here is the description of the used algorithm; we use the genetic algorithm from the Malva Library [16] with some custom modifications to support parallel execution.

Parallel algorithm: It uses the master-slave method. In each iteration the master generate threads that will evaluate the fitness function.

Multi-objective function: It optimize average bus speed (abs) and other vehicles average speed (avs), each one with a specific weight(x,y). In this case we use x=1 and y=1 to not give advantage to any of both.

$$F = x * avs + y * abs \qquad (1)$$

Chromosome representation: It's a vector of real numbers that represents traffic lights durations and also set the offset for each phase. Phase is the traffic light configuration for a particular intersection. The chromosome size depends on intersections size and phases on each intersection. We omitted

yellow lights as they not modify the real times.

It's important that the algorithm won't generate wrong solutions so we make sure to just modify durations and phases.

In the next figure we show the representation provided by SUMO, we can see how the phases are represented, for instance "GGGGrrGGGGrr" spent 52 seconds. "G" is Green, R is Red Y is yellow.

Offset is the starting time of the phase.

```
<add>
    <tlLogic id="cluster_-46_-48_-50" type="static" offset="0">
        <phase duration="52" state="GGGGrrGGGGrr" />
        <phase duration="3" state="yyyyrryyyyrr" />
        <phase duration="26" state="rrrrGGrrrrGG" />
        <phase duration="3" state="rrrryyrrrryy" />
    </tlLogic>
    <tlLogic id="cluster_1858033111_917381626_917457905" type="static" offset="1">
        <phase duration="44" state="rrrrrrrGGGGrrr" />
        <phase duration="3" state="rrrrrrryyyyrrr" />
        <phase duration="12" state="rrrrGrrrrrrGrr" />
        <phase duration="3" state="rrrryrrrrrryrr" />
        <phase duration="32" state="GGGGrrrrrrrrrr" />
        <phase duration="3" state="yyyyrrrrrrrrrr" />
    </tlLogic>
</add>
```

Figure 2: SUMO traffic lights representation

Crossover: It uses one-point crossover specific for this problem.

Mutation: For each cross it randomly change the traffic light duration given a range of numbers. Also it changes randomly the starting phase in the traffic light.

Selection and replacement: It replace parents per children, the parent selection is by tournament and children selection by roulette method.

```
1: Init( Pop(0))
2: generation = 0
3: while NOT Stop Criteria do
4:    Evaluate Pop(generation)
5:    Parents = Selection(Pop(generation))
6:    Children = Crossover(Parents) y Mutation(Parents)
7:    NewPop = Replace Pop(generation) with Children
8:    generation++
9: end while
10: return Best solution
```

Figure 3: Genetic Algorithm

5 Experimental Analysis

5.1 Execution platform

The platform used was the Cluster Fing [3], it's a set of independent computers connected to work as a single system, which give us high computer power. Hardware description:

- 9 servers

- Quad core Xeon E5430, 2x6 MB cache, 2.66GHz, 1.333 MHz FSB.
- 8 GB memory per node.
- 64 bits architecture.
- File Server: Two 1 TB disks.

- Communication Switch
  - Dell Power Connect, 24 Gigabit Ethernet ports.
- Switch KVM (16 ports)

5.2 Parameter configuration

We search the best initial parameter configuration doing experimental test with different combinations.

We create 3 traffic instances: low, middle, high and execute 3 simulations each, this way the algorithm is not skewed by a particular case. These are the parameters:

- **Simulation time:** We make sure that 80% of vehicles complete they trips, we found the simulation time of 4000 steps (SUMO internal metrics) that represent 66mins of real time.
- **Stop criteria:** We select the generation number, after doing test we found that 500 generations was a good number because the fitness graphic has no much change after 400 generations.
- **Population size**: We test population of 32, 48 and 64 having in mind that the max number of processors in the cluster is 64 per node, so we had that limit. The results shown that there is little difference between the population sizes, so we select 32 because it consume less resources.

Population	Execution Time (m)	Average fitness ± Standard deviation
32	4853	16.37 ± 0.5
48	6772	15.84 ± 0.3
64	10184	16.46 ± 0.6

Table 1: Population test

- **Mutation probability:** We test the values: 0.01, 0.05 and 0.1
- **Crossover probability:** We test the values : 0.5, 0.8 and 1

We test all mutation and crossover combinations getting the average fitness to create the next table.

pc	pm	Average fitness ± Standard deviation
0.5	0.01	16.09 ± 0.30
0.5	0.05	15.60 ± 0.17
0.5	0.1	16.16 ± 0.42
0.8	0.01	16.04 ± 0.55

pc	pm	Average fitness ± Standard deviation
0.8	0.05	15.85 ± 0.32
0.8	0.1	16.08 ± 0.34
1	0.01	16.08 ± 0.45
1	0.05	15.82 ± 0.34
1	0.1	16.04 ± 0.25

Table 2: Crossover probability (pc) and mutation probability (pm) combinations

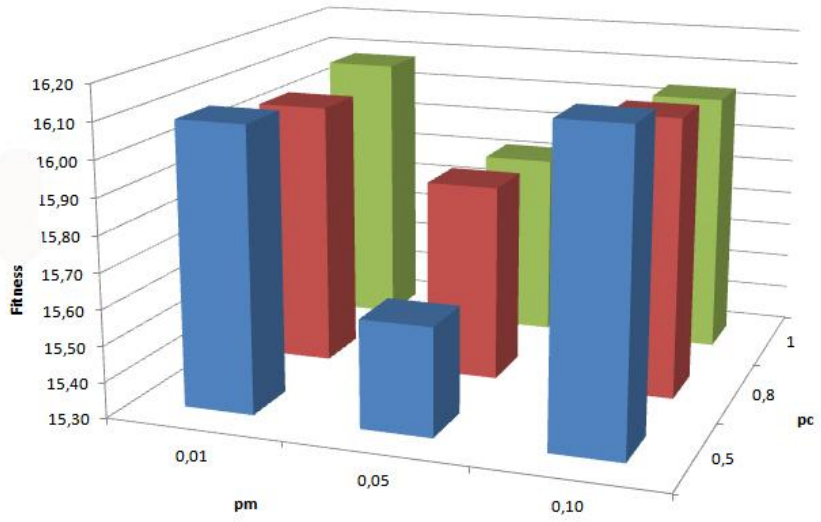

Figure 4: Crossover (pc) and mutation probability (pm) combinations

As we can see in the table and graphics for a pm=0.05 we get the worst results. Another interesting fact is that there is no big difference between the other combinations.

We tested that the each data follow a normal distribution so we can apply the Student-Test.

Having the test hypothesis:

H0: u1 = u2

H1: u1 <> u2

(With u1 group1 average and u2 group2 average)

If we compare the best average fitness (0.5-01) and the worst (0.5-005) with the Student-Test we get t(x) = 0.07 this means that for a significance level of 0.1 the null hypothesis is rejected so there is statistical evidence to choose the best average combination (0.5-0.1) over the worst (0.5-0.05) to execute the Algorithm.

To search for the best combination we select the two best averages to compare (0.5-0.1) and (0.5-0.01). In the Student Test we get t(x) = 0.71 that tell us that there is no statistical evidence between them so we can select either.

We select (0.5-0.01) as it has a good average fitness and also a low standard deviation.

5.3 Scenarios

These scenarios are the ones that are going to be simulated to get the data. In this case we run 21 independent runs on three different traffic car density (high, medium, low), the bus density is the same as there is no more frequencies but there is more time in the bus stops because there are more people.

High traffic: There are around 2800 vehicles and 70 busses

Medium traffic: 2000 cars, 70 busses

Low traffic: 1000 cars, 70 busses

Base Case: This represent the actual situation of traffic density, traffic light configuration, routes, etc. All of this data was gather in-situ or from public information.

Algorithm Case: This is when we execute the algorithm and get a new set of traffic light configuration

5.4 Results

Base Case result:

	Average bus speed (km/h)	Average car Speed(km/h)	Fitness
Low traffic	15.89	32.45	13.42
Medium traffic	14.59	28.81	12.05
High traffic	14.31	26.36	11.3

Table 3: Base case results

Algorithm result:

	Average bus speed (km/h)	Average car Speed(km/h)	Average fitness ± Standard deviation	Improvement (%)
Low traffic	17.92	34.3	14.50 ± 0.14	8
Medium traffic	16.95	33.29	13.95 ± 0.15	15.7
High traffic	16.51	32.90	13.72 ± 0.17	21,4%

Table 4: Algorithm case results

We apply the statistical significance criteria to validate the results and for each case it passed.

$$\left| f_{AVG}(A) - f_{AVG}(B) \right| > \max(std(f_A), std(f_B)) \quad (2)$$

So we can say that there is statistical evidence that the algorithm improves the results comparing with the base case.

For instance this means that the average bus trip time from start to end of the corridor (6.5km) in a

high traffic situation was in the base case 27mins, and after the algorithm it improves to 23.5 mins. And in the car case the time goes from 14.8 mins to 11.8mins.

And if we check the tables, the algorithm got for high traffic better values than the current low traffic times. So we can say that the algorithm change the traffic behavior, transforming the current high traffic speed average to a low traffic speed average.

Speedup

Here is the analysis of the speedup that we get. The time is the duration of the execution of 7 independent runs of the algorithm.

Cores	Time(m)	Speedup
1	11399	1
4	1936	5.8
8	1491	7.6
16	803	14.1

Table 5: Algorithm case results

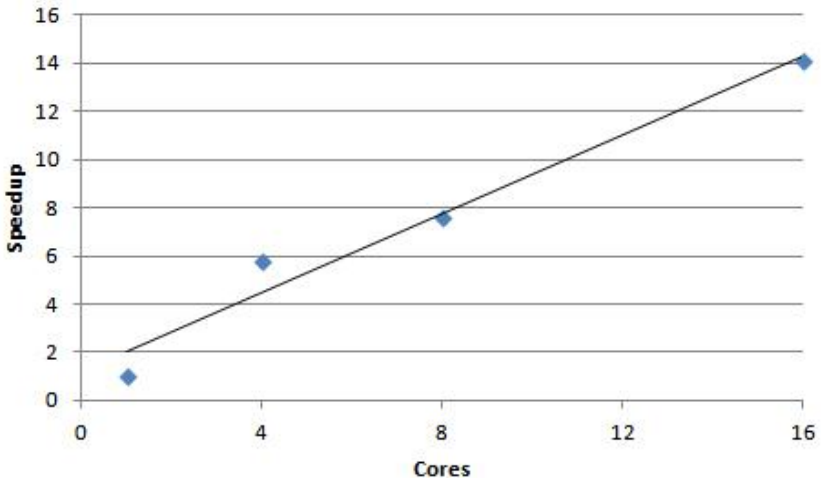

Figure 5: Speedup from 1 to 16 cores

As we can see in the table and graphic it has a near linear speedup behavior which demonstrates that it's a well scalable algorithm.

6 Conclusions

6.1 Conclusions

All goals from the beginning of the project were accomplished; we found good and useful information when doing the research that helped us to improve our solution. As we studied the traffic problem we realized that it really affects the people lives and the search of new solutions could improve the quality

of their lives.

One of the most tedious work was related to the map creation and simulation data, because we wanted to be very accurate so we put a lot of effort at gathering the data in situ, validating the results, asking experts, etc.

Even thought the synchronization light traffic is a hard problem to resolve, the results show good results when the algorithm it's executed.
The development of a parallel algorithm is critical in complex problems that require a lot of computer power like the one studied, as the execution time required is really high. We found that our algorithm has almost a linear speedup.

The same algorithm can be applied to other route networks just changing the input data: map, traffic, traffic light configuration and busses.

6.2 Future Work

It would be interesting to create a new alternative scenario by changing things that can improve the results, as reducing the number of bus stops, rules on the intersection, adding parallel routes, etc.
Also we could execute another test over the fitness function to know how it changes its behavior when we adjust the weights, for instance we can give more weight to bus traffic and this will lead (we believe) to an improvement in the bus speed.
The map development for the simulation could be a slow and error prone process, so in the future could be better to use tools that speed up the process.

The related work of Montana [7] and Vogel [19] suggest the use of real time adaptable algorithm; this could add new elements like sensors but also could improve the results.

References

[1] DiFebbraro, Giglio, and Sacco. On applying petri nets to determine optimal offsets for coordinated traffic light timings.

[2] CEPAL. Congestión de tránsito, el problema y cómo enfrentarlo. http://www.cepal.org/publicaciones/xml/9/13059/cue-87.pdf , 2003.

[3] Cluster Fing http://www.fing.edu.uy/cluster/index.php

[4] J. J. K. G. Y. Lim and Y. S. Hong. The optimization of traffic signal light using artificial intelligence. 10th IEEE Int. Conf.Fuzzy System

[5] IMM. Intendencia Municipal de Montevideo. http://www.montevideo.gub.uy

[6] S. López, P. Hernandez, A. Hernandez, and M. Garcia. Artificial neural networks as useful tools for the optimization of the relative offset set between two consecutive sets of traffic lights.

[7] D. J. Montana and S. Czerwinski. Evolving control laws for a network of traffic signals. 1996.

[8] J. Penner, R. Hoar, and C. Jacob. Swarm-based traffic simulation with evolutionary traffic light adaptation. 2002.

[9] A. K. Rathi. Urban network traffic simulation:traf-netsim program. Journal of Transportation Engineering, 116(6):734{743, November-December 1990.

[10] N. M. Rouphail, B. B. Park, and J. Sacks. Direct signal timing optimization: Strategy development and results. 2000.

[11] J. Sánchez, M. Galán, and E. Rubio. Genetic algorithms and cellular automata: A new architecture for traffic light cycles optimization. 2004.

[12] J. Sánchez, M. Galán, and E. Rubio. Applying a traffic lights evolutionary optimization technique

to a real case:”las ramblas" area in santa cruz de tenerife. 2008.

[13] J. Sánchez, M. Galán, and E. Rubio. Traffic signal optimization in “la almozara" district in saragossa under congestion conditions, using genetic algorithms, traffic microsimulation, and cluster computing. 2010.

[14] D. H. Stolfi. Optimizacion del trafico rodado en ciudades inteligentes. 2012.

[15] SUMO. Simulation of urban mobility. http://sumo.dlr.de/wiki/Main_Page

[16] OSM. Open street map. www.openstreetmap.org/

[17] The Malva project. http://themalvaproject.github.io/

[18] K. T. K. Teo, W. Y. Kow, and Y. K. Chin. Optimization of traffic flow within an urban traffic light intersection with genetic algorithm. 2010.

[19] A. Vogel, C. Goerick, and W. von Seelen. Evolutionary algorithms for optimizing traffic signal operation. 2000

[20] Ana Olivera interview 2015 (Montevideo Mayor). http://www.elpais.com.uy/informacion/ana-olivera-no-justificacion-corredor.html

[21] Ana Olivera interview 2013 (Montevideo Mayor) http://www.elpais.com.uy/informacion/garzon-olivera-admitio-errores.html