Applying a Traffic Lights Evolutionary Optimization Technique to a Real Case: "Las Ramblas" Area in Santa Cruz de Tenerife

Javier Sánchez, Manuel Galán, and Enrique Rubio

***Abstract*—In previous research, we have designed and successfully tested a Traffic Light Cycles Evolutionary Optimization Architecture. In this paper, we attempt to validate those results with a real-world test case. For a wide area in Santa Cruz de Tenerife city—Canary Islands—we have improved traffic behavior, using our optimized traffic light cycle times in a simulated environment. Throughout this paper, we present some of the experiences, knowledge, and problems encountered.**

***Index Terms*—Cellular automata (CA), combinatorial optimization, genetic algorithms (GAs), microscopic traffic simulator, traffic lights optimization.**

I. Introduction

THE PROBLEM OF traffic tie-ups in every major world city has caused not only inconvenience, but also a significant economic issue. If not managed correctly, it could seriously hinder the development of our cities.

The nonstopping overload process that traffic infrastructures are suffering calls for new solutions. In most cases, it is not viable to extend traffic infrastructures because of costs, limited available space, and environmental impact. Thus, we urgently need to optimize existing infrastructures to obtain the best possible service.

Many traffic management initiatives provide online traffic information directed at drivers via the Internet, commercial radio, Global System for Mobile Communication (GSM), or electronic panels. Some studies have demonstrated that if drivers had this sort of information, it would improve traffic flow.

Most research in this field has been aimed at solving small, specific problems using ad hoc methods. When more general or wider problems are faced, they are generally handled on a trial-and-error basis: somebody decides on traffic parameters and, depending on the results, some feedback corrections are applied.

This strategy has not changed significantly over time. Using simulators instead of real traffic tests as a feedback source implies great time savings. For example, microsimulators have proven fast and accurate. However, the method still depends on the engineers' experience and cannot ensure that the whole search space is covered.

One of the most important problems in traffic optimization is traffic light cycle[1] optimization. This is a challenging combinatorial problem which seems to have no known deterministic method to solve it at present.

Hence, it seems suitable to consider stochastic optimization methods such as genetic algorithms (GAs). Brockfeld *et al.* [1] demonstrate that traffic light cycles have a strong influence on traffic flow results. This is the reason we decided to take on this problem.

The rest of this paper is organized as follows. In Section II, we comment on the efforts of other researchers regarding traffic optimization. In Section III, we present our methodology. In Section IV, we show some previous experiences with this system. Section V describes the problem being addressed. Section VI enumerates the restrictions applied to solve the problem. In Section VII, we present our test results. Finally, in Section VIII, we set out some concluding remarks and project future research.

II. State-of-the-Art

There has been considerable research into traffic optimization. In this section, we provide some examples. In [2], an example of ad hoc architecture is used to optimize a nine-intersection traffic network. It uses GAs as an optimization technique running on a single machine. The CORSIM[2] model is used within the evaluation function of the GA but this work does not address scalability. Authors recognize that it is a customized nonscalable system. Our system is scalable thanks to the intrinsic scalability of the Beowulf cluster and the parallel execution of the evaluation function within the GA.

In [3], every intersection is optimized considering local information only. Moreover, it can be adapted to short- and long-term traffic fluctuations. In our case, we perform a global optimization instead of multiple local optimizations. We believe our approach exploits the traffic infrastructure more efficiently.

The "offset-time"[3] between two traffic lights is optimized using Artificial Neural Networks (ANNs) [4]. Although our system does not explicitly treat the offset time parameter, it deals with traffic optimization in a far more flexible manner.

Manuscript received November 16, 2005; revised July 4, 2006; November 27, 2006. This work was supported in part by the Ministerio de Educación y Ciencia of Spain (ref TSI2004-05949), which is financed 70% from European Union FEDER funds.

The authors are with the Innovation Center for Information Society (CICEI), Universidad de Las Palmas de Gran Canaria, Las Palmas 35017, Spain (e-mail: jsanchez@polaris.ulpgc.es; moreno@aha-dee.upc.es).

Digital Object Identifier 10.1109/TEVC.2007.892765

[1]Traffic light cycle: the finite sequence of states—e.g., green, orange, etc.—that a traffic light runs iteratively.

[2]CORSIM: Corridor traffic simulation model [5].

[3]Offset-time: The time from when a traffic light turns green until the next traffic light—for example, in a boulevard—also turns green.

In [6], Tveit, a Senior Researcher with *SINTEF*,[4] explains that a common cycle time[5] for a set of intersections is a less useful approach than a distributed, individualized one. His conclusions seem sound and convincing, so we consider them in our approach. In our system, every intersection has independent cycles.

In [7], a real-time local optimization of one intersection technique is proposed. It is based on fuzzy logic. Although an adaptive optimization may be very interesting—as we reported [8]—a global optimization seems to offer a more complete approach to the problem.

In [9], Petri Nets are applied to provide a modular representation of urban traffic networks. An interesting feature of this model is the possibility of representing the offsets among different traffic light cycles as embedded in the structure of the model itself. Even though it is a very interesting study, the authors only optimize the coordination among different traffic light cycles. Our cycle optimization methodology is completely flexible because we implicitly optimize not only traffic light offsets but also every *stage length*.

Li *et al.* [10] published another interesting paper using Petri Nets, where a Petri Nets-based approach to control a single intersection by programmable logic controllers (PLCs) is proposed. They compare three methods for modeling the traffic lights at an intersection and found the one that combines Petri Nets with PLCs was more suitable. Again, in this work only one intersection is optimized and not a whole traffic network.

Smith [11] favors the use of responsive signals,[6] with network capacity (rather than total travel costs) as a control criterion. Network capacity is maximized if the signals equalize traffic density on the most occupied parts of the network. This is another example of multiple local optimizations instead of a global optimization, as in our case.

Hong *et al.* [12] proposed the optimal green time algorithm concept. This reduces average vehicle waiting time, while improving average vehicle speed using fuzzy rules and neural networks (NNs). Through computer simulation, this method has been proven much more efficient than using fixed time cycle signals. The fuzzy neural network will consistently improve average waiting time, vehicle speed, and fuel consumption. This study only considers a very small amount of traffic signals—two intersections in the same area—for cycle optimization. We do agree with them about the nonsuitability of fixed cycles.

Spall *et al.* [13] present a NN approach for optimizing the traffic light cycle. A NN is used to implement the traffic lights control function. The training process of the NN is fed exclusively with real data. Thus, it would only be useful in systems with an online data acquisition module installed. However, so far, such systems are not common at all.

In [14], a fuzzy control system for extending or shortening the fixed traffic light cycle is suggested. By means of electrosensitive traffic lights, they can extend the traffic cycle when many vehicles are passing on the road or reduce the cycle if there is limited traffic flow. Through simulation they presented efficiency improvement results. This work performs a local adaptation for a single traffic light instead of global optimization.

In a very interesting paper, Wiering *et al.* [15] describe traffic as a set of intersections optimized in a standalone manner. They recommend using reinforcement learning algorithms to optimize what they consider a multiagent decision problem. We do not agree, although a local optimization can obviously reduce average vehicle waiting times—this seems to happen with simulated tests in this work—we think a global optimization, taking into account every intersection in a designated area, would be more profitable.

III. Methodology: Description of Our Architecture

The architecture of our system comprises three items, namely, GA as nondeterministic optimization technique, cellular automata (CA)-based traffic simulator inside the evaluation routine of the GA, and Beowulf cluster as MIMD multicomputer. Through this section, we broadly describe the GA and the CA traffic simulator. Finally, a brief description of the Beowulf cluster will also be provided.

A. Genetic Algorithm (GA)

In this section, we will describe the GA employed.

1) Optimization Criterion. Fitness Function: After testing several criteria, we found the best results using the absolute number of vehicles that left the traffic network during the simulation.

During the traffic simulation, many new vehicles are created as if they were arriving at the inputs of the network. Moreover, during the simulation, many vehicles reach their destination point and, consequently, leave the network. The number of vehicles that reach their destination point easily illustrates the simulation and helps us to compare it with others.

So far, we have tested some other optimization criteria.

- Mean time at the network—mean elapsed time (MET). During the simulation, the arrival and departure time of every vehicle is stored. With these values we can easily calculate the number of iterations (seconds) spent by any vehicle leaving the network. Once the simulation finishes, the average time at the network is calculated.
- Standard deviation values of vehicle time at the network.
- A linear combination between the MET and the standard deviation of vehicle times at the network.
- A linear combination between the MET and the total number of vehicles that have left the network during the simulation.
- The traffic network mean occupancy density. To calculate this parameter, we divided the network into small sections and counted the number of vehicles inside every section.

In our search for our systems optimization criteria, we encountered an unexpected problem. If we included the minimization of the MET in the multicriteria objective function, we induced a very undesirable effect. The chromosomes that blocked the network faster were the best marked. This is because only a few vehicles were able to leave the network (in a limited amount

[4]SINTEF means The Foundation for Scientific and Industrial Research at the Norwegian Institute of Technology.

[5]Common cycle time: This is a very simple way of programming traffic lights in an intersection or groups of intersections. All the traffic lights share a cycle length. The starting point of each one of the states or *stages* in the particular cycle of every traffic light may be different, but the cycle period is the same for all of them.

[6]Responsive signals: Traffic signals capable of adapting their state to the current traffic situation near them.

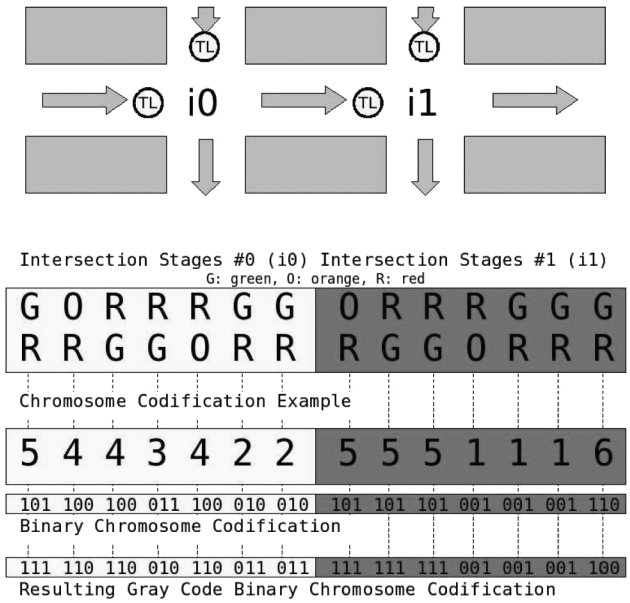


Fig. 1. Chromosome encoding.

of iterations) before it blocked. Hence, we obtained very low values for this criterion. Thus, we abandoned this criterion.

2) Chromosome Encoding: In Fig. 1, we present the chromosome encoding currently used in our system. In this figure, we are representing a chromosome example for a very simple traffic network case. It consists of two intersections and two traffic lights for each intersection.

Below the traffic network, we have put the stages[7] of each traffic light separated in two different color regions, one for each of the two intersections. The traffic light state at each *stage* may be green (G), orange (O), or red (R). This *stage* sequence is preestablished and will cycle *ad infinitum*—or until we stop the corresponding simulation.

Under the stage sequence a hypothetical chromosome is shown. Every integer means the time (seconds) of the corresponding cycle *stage*. The objective of our system is to optimize the length of each *stage* in order to get the very best traffic behavior from the network under study.

As one can see in Fig. 1, after several translation steps we obtain a binary Gray Code encoding (Black—[16]). We have shown this methodology to be very efficient for our case in [17]. We will explain this matter in more depth in Section IV-D, and in Appendix IV, we briefly explain how Gray Code works.

We use Gray Code because it is designed so that when a bit changes its value—when mutation occurs—the *stage length* value only increases or decreases one unit. This is a desirable feature because it makes the search space conform to the "Hamming Distance Metric."

3) Initial Population: On creating every individual in the initial population, we set a time range for the length of every preestablished *stage* and we choose a random value within this time interval.

For our study, we are including nine individuals in the initial population that are not created randomly; rather, these individuals are "solutions" provided by the City Council, as we will explain further in Section V-C.

a) Random Number Generation: For the random number generation, we have employed the MT19937 of Makoto Matsumoto and Takuji Nishimura, known as the "Mersenne Twister" generator. It has passed the DIEHARD statistical tests [18]. The seeds for this algorithm were obtained from the "/dev/urandom" device provided by the Red Hat 9 operating system.

4) Selection Strategy: We have chosen a truncation and elitism combination as selection strategy. This means that at every generation a small group of individuals—the best two individuals in our case—is cloned to the next generation. The remainder of the next generation is created by cross-overing the individuals belonging to a best fitness subset—usually, 66% of the whole population.

This combination seems to be the most suitable for our problem among a set of selection strategies tested. However, we do not rule out the possibility of changing it if better results seem attainable.

Other selection strategies are succinctly explained as follows.

- Elitism: The population is ordered by fitness. Then, a few with the higher fitness values (elite) are cloned to the next generation.
- Truncation: The population is ordered by fitness. Then, the population is divided into two sets, one to survive—with the higher fitness values—and another is simply discarded.
- Tournament: Small groups of individuals are chosen at random. The best fitness individual for each is selected.
- Random Tournament: This is similar to the Tournament Selection but here the best individual is not always selected. It will depend on a probability value.
- Roulette Linear Selection: Every individual has a survival probability proportional to its fitness value.
- Elitism plus Random Tournament.

5) Crossover Operator: We have tested some different crossover operators (uniform crossover, two point crossover at fixed points, and two point crossover at random points) and concluded that, at least for our work, the best was the third, in terms of behavior.

For a couple of parents, it simply chooses two random points at each one of the two chromosomes, cuts them into three pieces, and then interchanges the central chunk of them.

The way this operator is affected by using an integer or Gray Code encoding will be detailed in Section IV-D.

6) Mutation Operator: The value of a randomly chosen bit in the chromosome is just flipped.

The mutation probability is not fixed. We have also tested this and, in our case, results suggest the use of a variable mutation probability to our problem. It starts with a very high probability that will be decreasing multiplied by a factor in the range (0,1) until it reaches probability values near the inverse of the population size when approaching the end of the planned number of generations.

B. Traffic Simulator

Traffic simulation is known to be a very difficult task. There are mainly two different traffic model flavors. The first one is the macroscopic model family. They are based on fluid dynamics,

[7]Stage: Every one of the states associated to an intersection that contains a set of traffic lights.

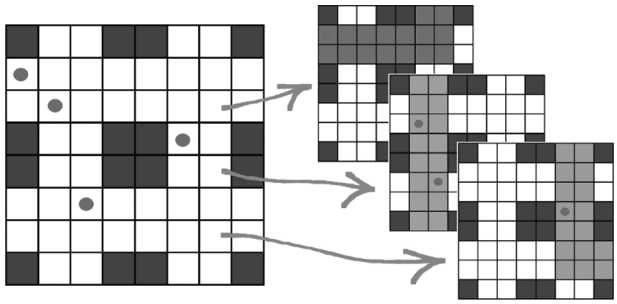


Fig. 2. Paths in our improved CA model.

since they consider traffic flow as a continuous fluid. On the other hand, we have the microscopic simulators. For them, traffic is considered as a collection of discrete particles following some rules. In the past decade there has been a common belief about the better performance of microscopic approaches for traffic modeling. One such widely used approach is the CA model.

There has been a long tradition of macroscopic approaches for traffic modeling. In the 1950s, some "first-order" continuum theories of highway traffic appeared. In the 1970s and beyond, some other "second-order" models were developed to correct their deficiencies. References [19]–[24] may illustrate some of these models. However, in [25], "second-order" models are questioned due to some serious problems, i.e., negative flows predictions and negative speeds under certain conditions.

Nowadays, microscopic simulators are widely used. One reason is that macroscopic simulators cannot model the discrete dynamics that arise from the interaction of individual vehicles [26]. CA seem to be faster than any other traffic microsimulator [27] and, as said in [28] "the computational requirements are rather low with respect to both storage and computation time making it possible to simulate large traffic networks on personal computers."

1) CA as an Inspiring Model: CA simulators are based on the CA Theory developed by von Neumann [29] at the end of the 1940s at the Logic of Computers Group of the University of Michigan. CA are discrete dynamical systems whose behavior is specified in terms of local relation. Space is sampled into a grid, with each cell containing a few bits of data. As time advances, each cell decides its next state depending on the neighbors state and following a small set of rules.

In the Cellular Automata model not only space is sampled, but also time and speed. Time becomes iterations. A relationship between time and iterations is established, e.g., $1(\mathrm{sec.}) \equiv 1(\mathrm{iteration})$. Consequently, speed turns into "cells over iterations."

In [30], we can find a well-described list of microscopic models and a comparative study of them. Although conclusions are not definitive, this work seems to demonstrate that models using less parameters have a better performance.

We have developed a traffic model based on the SK[8] model [31] and the SchCh[9] model [32]. The SchCh model is a combination of a highway traffic model—Nagel–Schreckenberg [33]—and a very simple city traffic model—Biham–Middleton–Levine [34]. The SK model adds the "smooth braking" for avoiding abrupt speed changes. We decided to modify our model inspired by the SK model due to its improved results for all the tests shown [30].

2) Our Improved CA Model: We have developed a nonlinear model for simulating traffic behavior that is based on the CA model. The basic structure is the same as the one used in CA. However, in our case, we add two new levels of complexity by creating two new abstractions: "Paths" and "Vehicles."

"Paths" are overlapping subsets included in the CA set. There is one "Path" for every origin-destination pair. To do this, every "Path" has a collection of positions and, for each one of them, there exists an array of allowed next positions. In Fig. 2, we try to illustrate this idea.

"Vehicles" consists of an array of structures, each one of them having the following properties.

1) Position: where the cellular automaton is located. Note that every cell may be occupied by one and only one vehicle.
2) Speed: the current speed of a vehicle. It means the number of cells it moves over every iteration.
3) Path: In our model, every vehicle is related to a "path."

These are the rules applied to every vehicle.

1) A vehicle ought to accelerate up to the maximum speed allowed. If there is no obstacle in its way (another vehicle, or a red traffic light), it will accelerate at a pace of one cell per iteration, every iteration.
2) If a vehicle can reach an occupied position, it will reduce its speed and will occupy the free position just behind the preceding.
3) If a vehicle reaches a red traffic light, it will stop.
4) Smooth braking: Once the vehicle position is updated, then the vehicle speed is updated too. To do this, the number of free positions from the current position ahead is taken into account. If there is not enough free space for the vehicle to move forward on the next iteration going at its current speed (hypothetically, since in the next iteration the traffic situation may change), it will reduce its speed by one unit.
5) Multilane traffic: When a vehicle is trying to move on, or update its speed, it is allowed to consider positions on other parallel lanes. For every origin-destination couple (path), for every cell there is a list of possible subsequent positions. The first considered is the one straight forward. If this is not free, there may be more possible positions in parallel lanes that will be considered. Of course, this list of possible next positions is created taking the basic driving rules into account.

Using these rules we can simulate many different path vehicles running in the same network. This model may be seen as a set of N_{paths} traditional CA networks working in parallel over the same physical points.

Note that so far, we are not considering different behavior for the green and the orange state. However, our architecture is designed in such a manner that we can modify this whenever we want to, with little effort.

C. Beowulf Cluster

The hardware of our system is based on a five node Beowulf cluster, due to its very interesting price/performance relationship and the possibility of employing Open Software

[8]Stephan Krauss, the author.

[9]Andreas Schadschneider and Debashish Chowdhury, the authors.

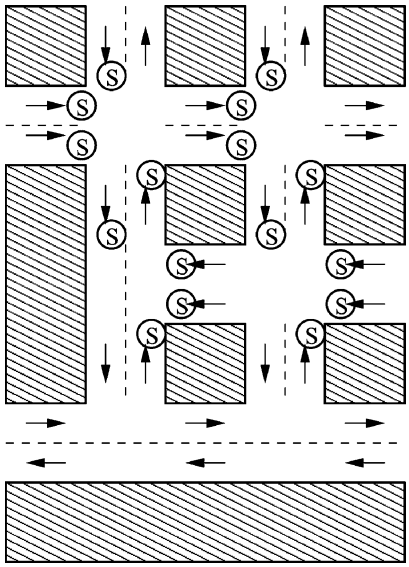


Fig. 3. Size_1 traffic network.

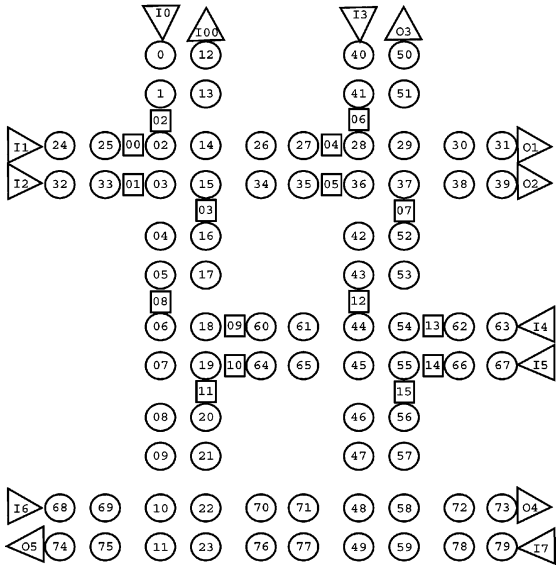


Fig. 4. Size_1 discretized traffic network.

on it. Moreover, this is a very scalable MIMD computer, a very desirable feature in order to solve many sorts—and scales—of traffic problems.

Every cluster node consists of a Pentium IV processor at 3.06 GHz with 1 GB DDR RAM and 80 GB HDD. The nodes are connected through a Gigabit Ethernet Backbone. Every node has the same hardware, except for the master node having an extra Gigabit Ethernet network card for "out world" connection.

Every node has installed Red Hat 9 on it—Kernel 2.4.20–28.9, glibc ver. 2.3.2 and gcc ver. 3.3.2. It was also necessary for parallel programming the installation of LAM/MPI (LAM 6.5.8, MPI 2).

In our application there are two kinds of processes, namely, *master* and *slave*. There is only one master process running on each test. At every generation it sends the chromosomes (MPI_Send) to the slave processes, receives the evaluation results (MPI_Recv) and creates the next population. Slave processes routine is an endless loop. They are always waiting to receive a new chromosome (MPI_Recv). Then, they evaluate it and send the evaluation result (MPI_Send).

IV. Experiences Carried Out With This Architecture

A. First Publication

In [8], we presented our architecture for the optimization of Traffic Light Cycles in a Traffic Network. It was based somewhat on the same three points of this study. The good results of a parallel speedup study convinced us that it was advisable to use a Beowulf cluster.

After comparing our results with those from other methodologies, we made our initial improvements on the mean time spent by each vehicle inside the network.

For that study, we used a very simple traffic network (Fig. 3) that once discretized, appears as in Fig. 4. In this representation, we are using the following symbols: a square meaning a traffic light position, a triangle meaning a traffic input or output, and a circle meaning a valid vehicle position. We will keep this set of symbols throughout the remaining figures.

TABLE I
Three Networks Scales Statistics

Scale	Points	T. Lights	Intersections	Inputs	Outputs	Chromosome Size
Size_1	80	16	4	6	8	96 bytes
Size_2	202	24	12	14	14	192 bytes
Size_3	248	40	20	18	18	320 bytes
Size_4	1176	160	80	36	36	640 bytes

B. Scalability Study

In OPTDES IV,[10] we presented a scalability study with our architecture. For four different network scales—their statistics are represented in Table I—we studied the execution time and the fitness evolution. In this table, "Points" means the number of cells of the respective network, with a rate of one sample every 7 meters (approximately), which is the average space needed by a vehicle in a traffic jam. "T.Lights" represents the number of traffic lights optimized. "Intersections," "Inputs," and "Outputs" mean the number of intersections, inputs and outputs of the respective network. Finally, "Chromosome Size" means the size (bytes) of every chromosome.

In that research, we found out that our system worked well for all cases, except perhaps for the Size_4 network. In this case, it would only be necessary to extend the hardware because of computational overload.

C. Deterministic Simulator Suitability

In paper [35], we compare our deterministic traffic simulator with a stochastic one. There are three differences between a standard stochastic simulator and our deterministic version.

1) The cells updating order. In the stochastic version, the order of the cells to be updated is chosen at random. In our deterministic version, we have written a recursive function to calculate the dependencies graph. With it, we prepare a nondeadlocking updating order at the beginning of the simulation.

[10]Optimization and Design in Industry IV, Tokyo, Japan, September 26–30, 2004.

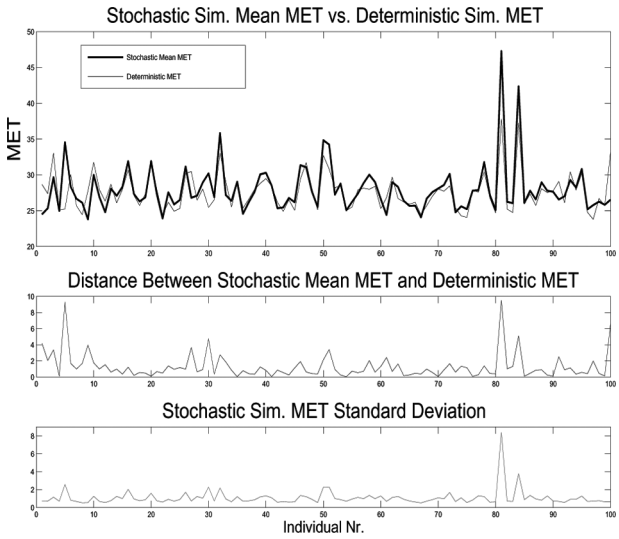


Fig. 5. Size_1 network MET comparison.

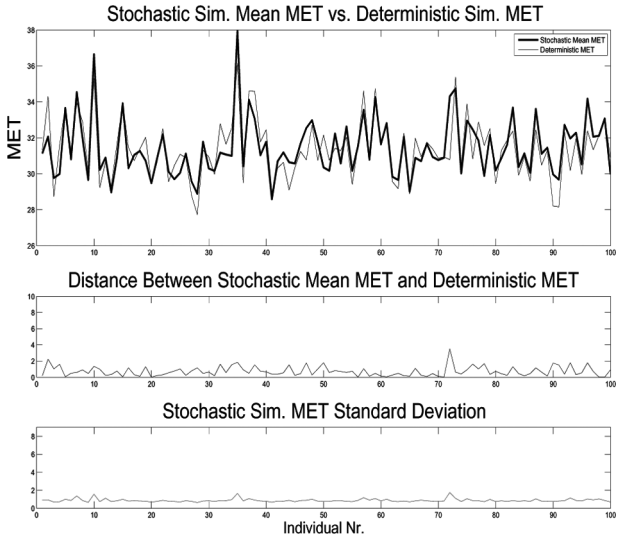


Fig. 7. Size_2 network MET comparison.

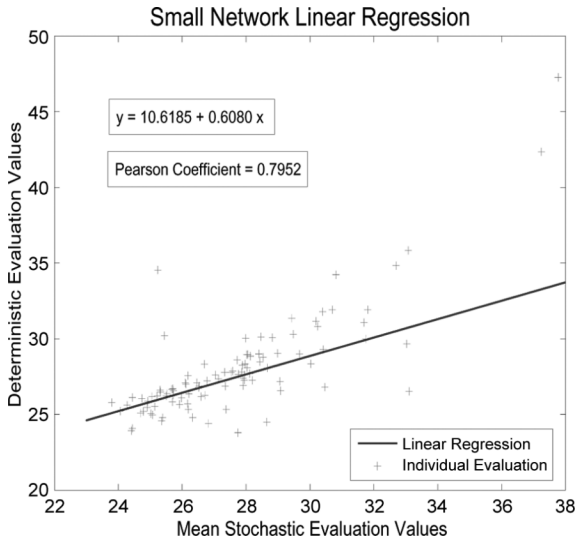


Fig. 6. Size_1 network linear regression.

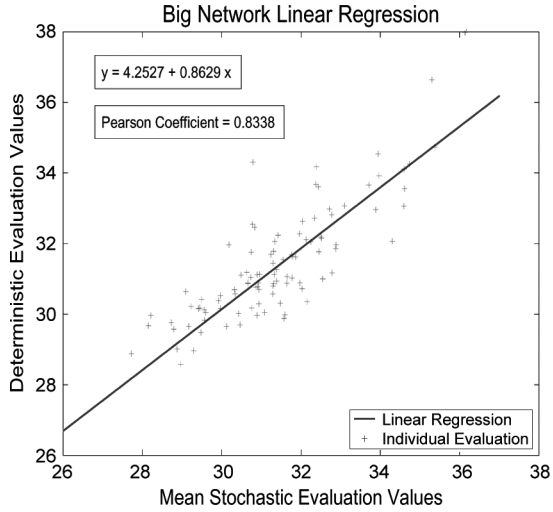


Fig. 8. Size_2 network linear regression.

2) The new vehicle creation time. In the stochastic version, every input has a probability of new vehicle arrival. So, the creation time of new vehicles depends on a random variable. In the deterministic case, we create new vehicles at fixed periods, proportional to the real traffic flow at every input.
3) The acceleration probability. In the deterministic case, when updating every vehicle speed, if it has free way and is under the maximum allowed speed it will always accelerate. However, for the stochastic case, there is an acceleration probability—usually greater than 0.7. So, vehicles may accelerate or not.

In the same study, we demonstrated that we obtain similar results using a standard stochastic simulator and a deterministic one. Furthermore, by using the deterministic version, we got huge computing power savings. This is because if we use a stochastic simulator, we need to run lots of simulations to get consistent values for the fitness of every population individual.

In that work, we tested this by using three different traffic networks—Size_1, Size_2, and Size_3 from Table I. We ran 1000 stochastic simulations and one deterministic simulation for every individual.

We used a population of 100 individuals and every traffic simulation ran through 1500 iterations.

We present a set of six graphs—Figs. 5 and 6 for the Size_1 network; Figs. 7 and 8 for the Size_2 network; and Figs. 9 and 10 for the Size_3 network. In Figs. 5, 7, and 9 the following items are represented.

1) In the first row, we draw the MET—elapsed time from when a vehicle reaches the network until it leaves—for both simulators. Note that in the stochastic case the average value of all the executions is represented.
2) In the second, the distance between the average values of the stochastic simulator and the deterministic simulator value is represented.
3) Finally, we plot the standard deviation of the stochastic simulator values.

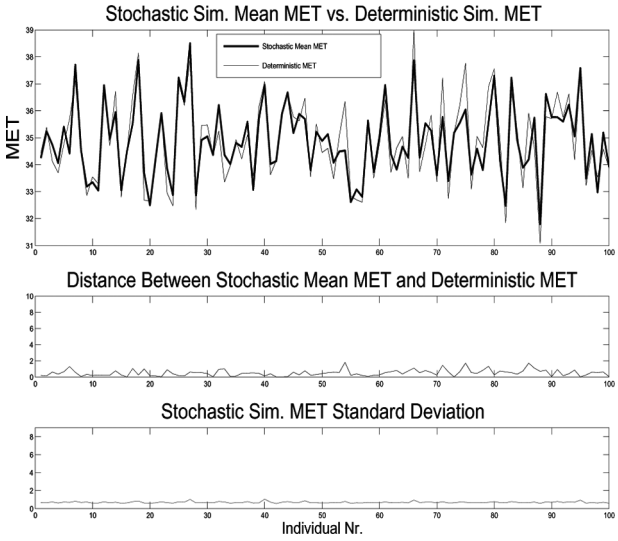


Fig. 9. Size_3 network MET comparison.

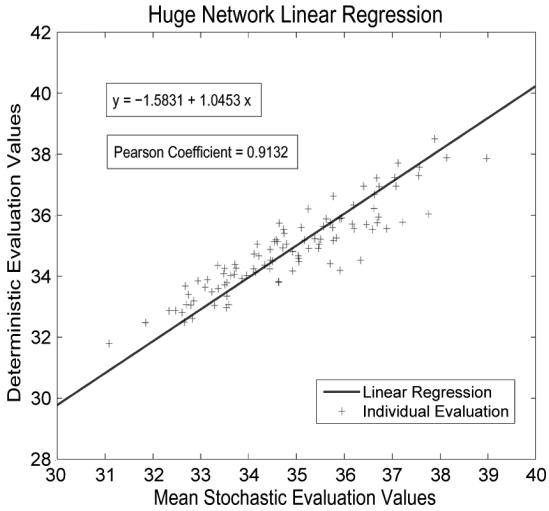


Fig. 10. Size_3 network linear regression.

From these graphs one may observe that the two main plots—mean stochastic and deterministic values—are highly correlated.

In Figs. 6, 8, and 10, we plot the linear regression function relating both kinds of simulators. Besides, the individual evaluation values are represented with plus signs.

We also include some other interesting statistics in Table II. In the first column, we have the network scale of the test. In the second column, we have the Pearson correlation coefficient [(1)]

$$\rho = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \overline{y})^2}}. \tag{1}$$

The third column displays the mean euclidean distance (MED) between the individual evaluations and the regression function. Finally, we have the mean computational cost ratio (MCCR), which is the average value in the whole population of the ratio for each individual between the execution time of the stochastic simulations (the sum of 1000 simulations) over the deterministic simulation time. For the MCCR, we have used

TABLE II
STATISTICS TABLE FOR THE DETERMINISTIC VERSUS STOCHASTIC SIMULATOR STUDY

Scale	ρ	M.E.D.	M.C.C.R.
Size_1	0.7952	2.4595	63.90
Size_2	0.8338	1.6786	74.49
Size_3	0.9132	1.7214	70.87

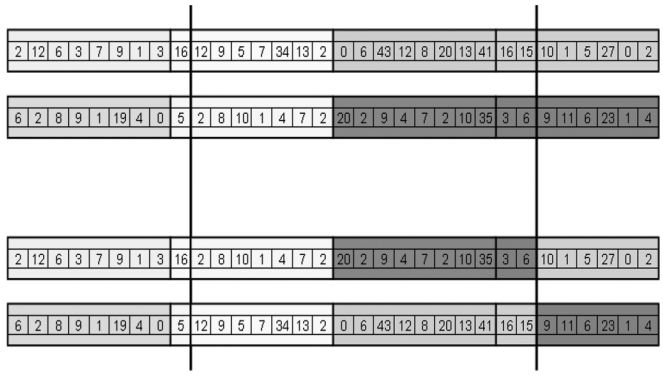


Fig. 11. Gene level two point crossover.

(2). In this equation N_{pop} means the population size. T_i^s means the accumulated execution time for the 1000 stochastic simulation run and T_i^d means the execution time for the deterministic simulator, both of them for the ith individual

$$\text{MCCR} = \frac{\sum_{i=1}^{N_{\text{pop}}}\left(\frac{T_i^s}{T_i^d}\right)}{N_{\text{pop}}}. \tag{2}$$

The conclusions from this work may be summarized in two points.

1) We confirmed that the stochastic simulator is a convergent process well suited to be used as a reference metric for the optimization.
2) We demonstrated that the deterministic simulator outputs are highly linearly correlated with the stochastic ones.

So with our deterministic simulator, we can arrange the population ranking in order of fitness at least as well as with the stochastic simulator, but with a remarkably lower computing time.

D. Bit Level Crossover

In this work [17], we describe the difference between the two types of genetic encoding studied, which yield different crossover and mutation strategies.

In the first case, we are using an integer encoding. In Fig. 11, we show a case of a two point crossover between two chromosomes corresponding to a very simple traffic network: four intersections each one of them having eight *stage* cycle lengths. We must remember that every integer means the time (in seconds) of the corresponding cycle *stage*. In this case, each integer is a *gene* with an *atomic* meaning.

In general, the crossover and mutation operator could be applied considering this integer (or gene level) encoding (see Fig. 11). For example, when a gene level crossover occurs, two individuals interchange their genes without modifying them. It means that the only change in genes is a consequence of

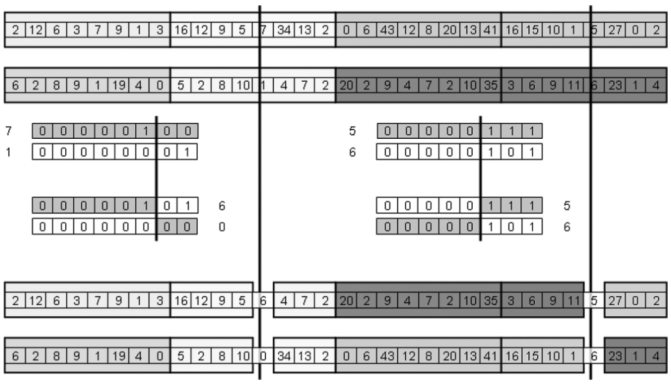


Fig. 12. Bit level two point crossover.

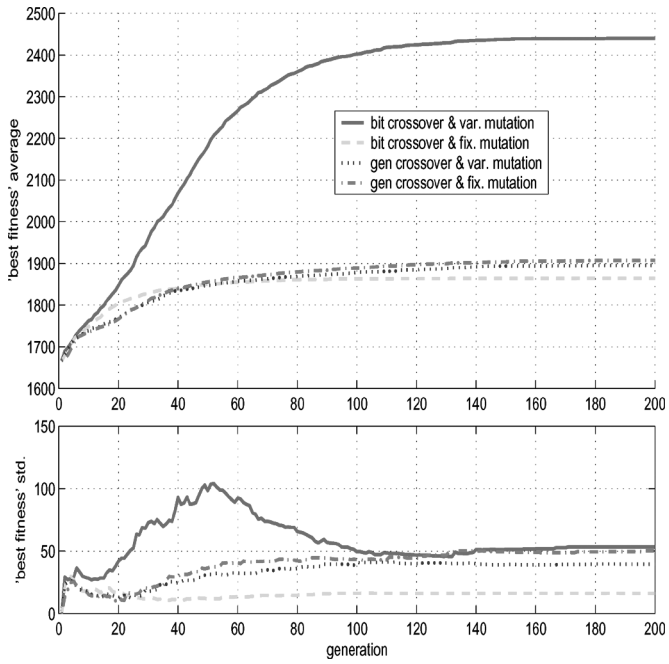


Fig. 13. Bit level versus gen level crossover fitness evolution comparison.

mutation. The gene level mutation will just change an integer value for another one (within its valid range).

On the other side, we may choose a bit level crossover and mutation operators. We represent the bit level crossover in Fig. 12. In this case, every stage duration is coded—and treated—using the Gray binary code ([16]). Due to the importance of Gray Code in genetic encoding, we explain it in further detail in Appendix IV.

When a bit level crossover occurs, it is obvious that the gene content is modified. It causes the population genes to change frequently.

In Fig. 13, we have represented the "best fitness" mean evolution for each case. Thus, we stored the best fitness value, generation by generation, for every execution of the GA. Then, we calculated the mean curve considering the 30 executions we ran for every test case. We also calculated the standard deviation of those values for every test case—in the bottom of Fig. 13.

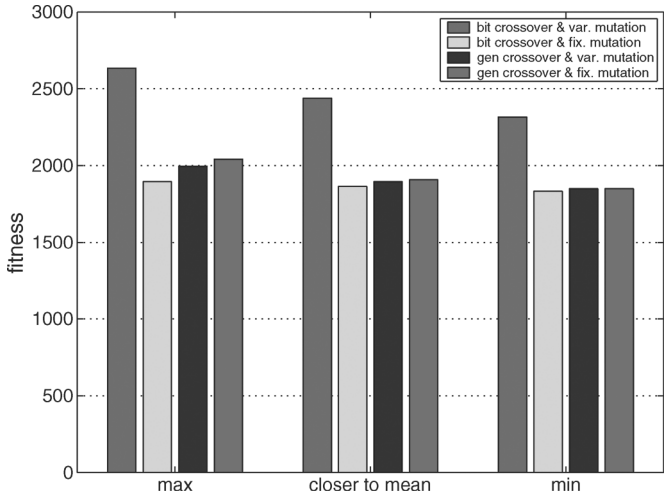


Fig. 14. Bit level versus gen level crossover relevant fitness values.

Note that for the couple consisting of a bit level crossover and a variable mutation probability (explained in Section III-A6), we obtained the best results. Furthermore, it seems that premature convergence occurs—found a local maximum—for the other three cases.

In Fig. 14, we depict for every combination the maximum, the minimum, and the closest to mean fitness values.

The main topic in this work was to demonstrate—by means of a wide set of tests—that at least in our particular case, a bit level crossover combined with a variable mutation probability may provide a great saving of computing time. Besides, we have seen that this choice lets the algorithm cover the solution space faster due to a bigger gene variability between generations. This combination seems to avoid premature convergence.

We should indicate that a Gray Code was specifically chosen because it is designed in such a way that when a bit changes its value—when mutation occurs—the *stage length* value only increases or decreases one unit.

V. Test Case Description

A. "Las Ramblas" Zone in Santa Cruz de Tenerife

Santa Cruz de Tenerife is a port city situated on the northeast coast of Tenerife Island. Tenerife is the biggest and most populated island in the Canary Islands, a Spanish archipelago off the Northwest Africa. Santa Cruz de Tenerife is the islands capital city. Situated at 28.47° north latitude and 16.25° east longitude, the city covers is 150.56 km^2 and boasts a population of 221,567 inhabitants—ISTAC[11] 2005 data. The city is the most important business center in Tenerife.

The city's economy is mainly service-based, with the exception of some chemical companies and oil refineries.

In the central district of "Las Ramblas," there is a high concentration of stores and also some shopping centers. Hence, we chose a heavily loaded traffic zone which is most suitable for

[11]Instituto Canario de Estadística.

Fig. 15. The Canary Islands with Santa Cruz de Tenerife highlighted.

testing our system. The high economic activity of the area makes every traffic improvement very profitable.

B. Traffic Department Collaboration

Once we decided to work with this city, our group contacted the Traffic Department Chief for the City Council, Mr. Hilario Rodríguez González. We explained our research to him and he was eager to collaborate with us and provided all the necessary data.

We requested three kinds of data. First, we needed topological data of the desired zone.

We also required an origin-destination probability matrix which is useful for simulating what would be the destination of every new vehicle arriving in the area under study. They could not provide such data directly, but were able to supply us with many traffic statistics, letting us build the aforementioned matrix ourselves—assuming some simplifications, as explained in Section VI.

We also needed those statistics for simulating, as close to actual traffic as attainable, the traffic arriving at the network.

Finally, we needed the current traffic light programming to perform a comparison between our optimized results and theirs.

C. Data Preprocessing

Throughout this section we will explain how we prepare the supplied data for feeding our optimization engine. In Fig. 16, we show a sample of the topology discretization for the supplied map. We set a point approximately every 7 meters—the usual distance needed by a generic vehicle in a traffic jam.

We extend this topology discretization to the zone in Fig. 17, yielding the discretized network of Figs. 18 and 19. In these figures, we only show the scale of our problem. It includes 1643 cells, 42 traffic lights, 26 inputs, and 20 outputs.

The Traffic Department of Santa Cruz de Tenerife City Council kindly provided us with nine sets of solutions used at that time for the studied network. The first one is labeled LC, which means that this combination is applied whenever the communication with the Control Center is lost. The others are labeled R0 to R7.

D. Objective Formulation

For a city zone, presented in Fig. 17, once discretized into a set of points as plotted in Figs. 18 and 19, taking into account the restrictions specified in Section VI and the probability matrix of Appendix III-A, our aim is to calculate the optimal times for each *stage*. Those times should lie within a specified range. These ranges may be set up by default or according to any special requirements.

Finally, to solve this problem, we include into the initial population nine chromosomes containing, respectively, the nine *stage length* sets currently used in the zone. We do this to ensure better results than the ones obtained with the supplied combinations.

VI. Problem Constraints

In this section, we set out the constraints applied in solving the formulated problem.

A. Low Traffic Streets

Our model depends on a close-to-reality simulation of traffic behavior in the analyzed network. This means that we need accurate statistics of what is happening on the streets to achieve the following two objectives.

1) Obtain a sound origin-destination probability matrix, which would let us estimate the destination for every new vehicle created at each input.
2) Simulate the real traffic behavior and the one using our results.

Hence, the more accurate traffic statistics available, the lesser the assumed simplifications. In Appendix I, the considered and deprecated streets for this work are specified. In Appendix III, there is a list of the simplifications assumed for building the origin-destination matrix.

B. Optimized Traffic Lights

So far, we do not consider pedestrians in our model. The main objective of our optimization is to maximize the traffic through the network. Further research will consider external interruptions of traffic provoked not only by foreseeable agents such as pedestrians, trains, tramways, etc., but also by unexpected issues such as accidents, demonstrations, and so on.

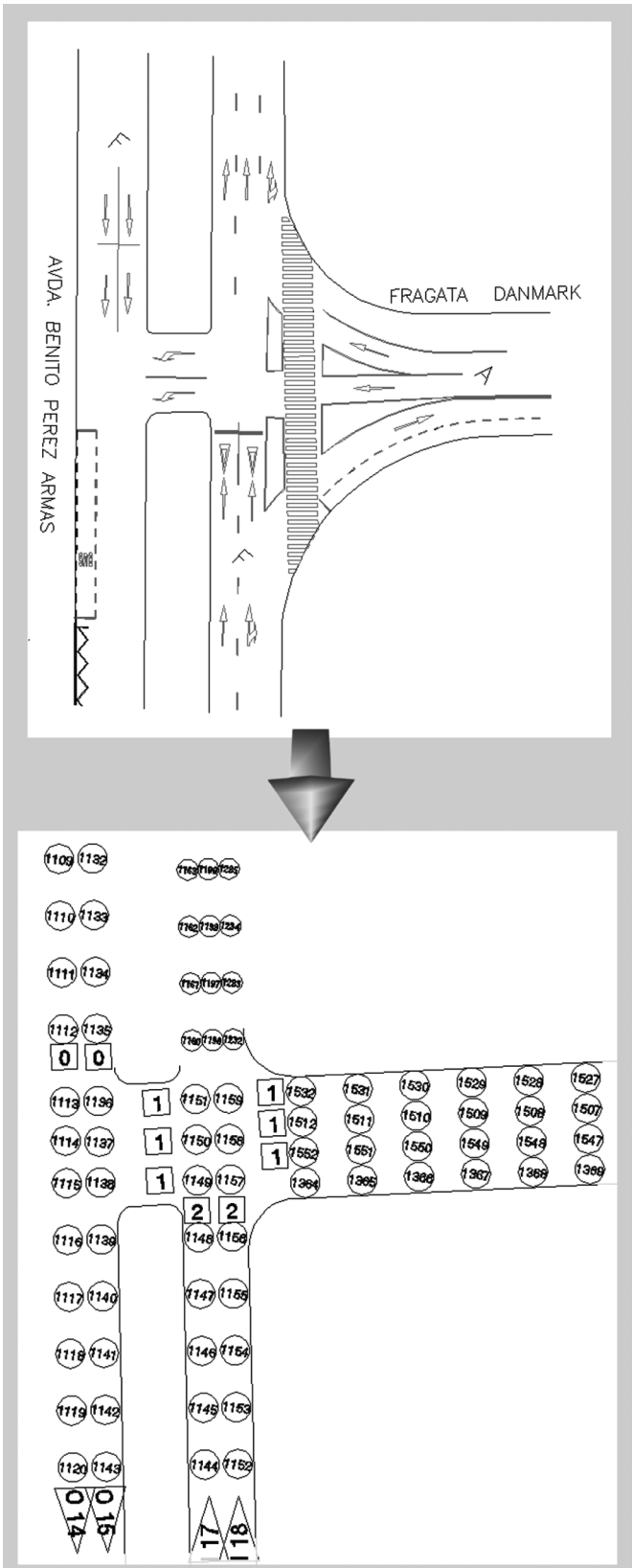


Fig. 16. Topology discretization.

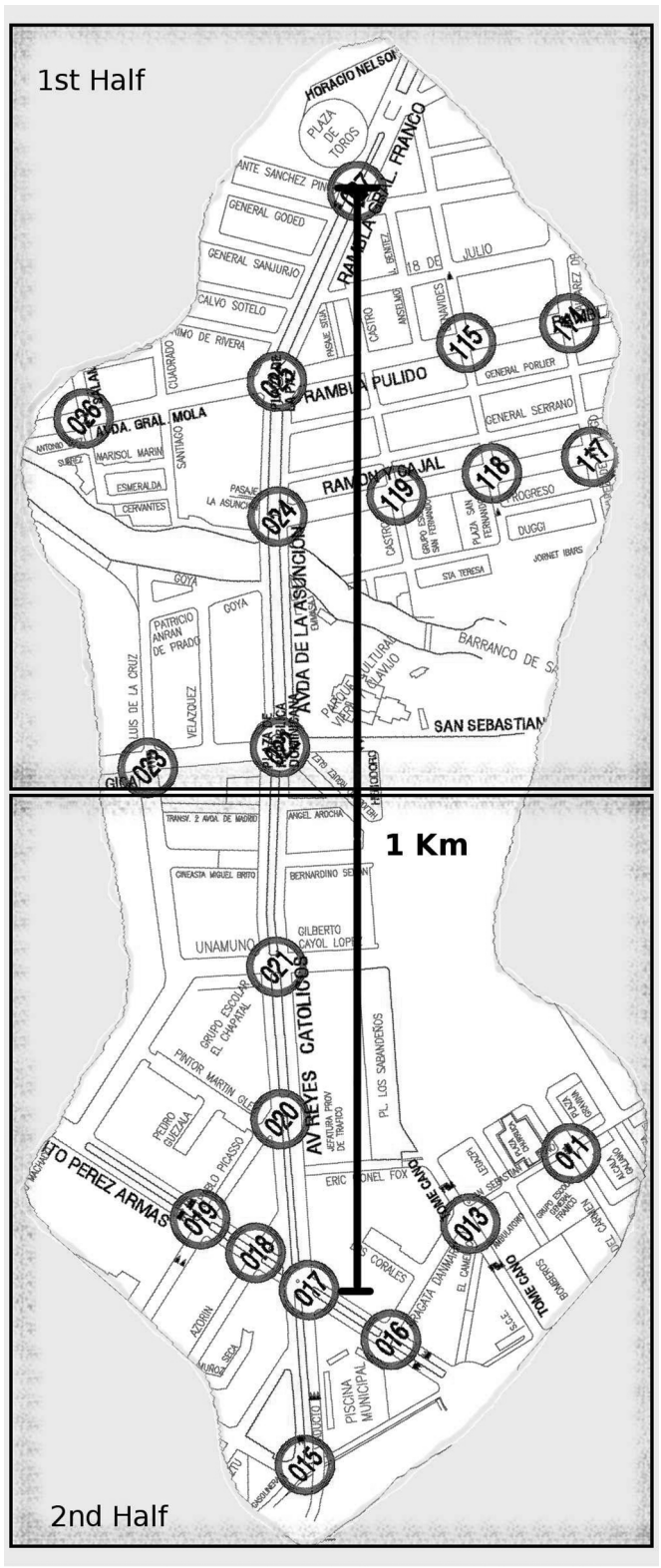


Fig. 17. "Las Ramblas" zone of Santa Cruz de Tenerife.

A key factor for any Advanced Traffic Management System is the optimization of the pedestrian network minimizing, for example, walking time. A remarkable goal would be to decouple the pedestrian network from motorized network by pedestrianized streets, bridges, or tunnels. However, as motorized traffic has more significant economic and environmental consequences, we have opted for optimizing traffic lights as a first approach to a global and complex problem. Therefore,

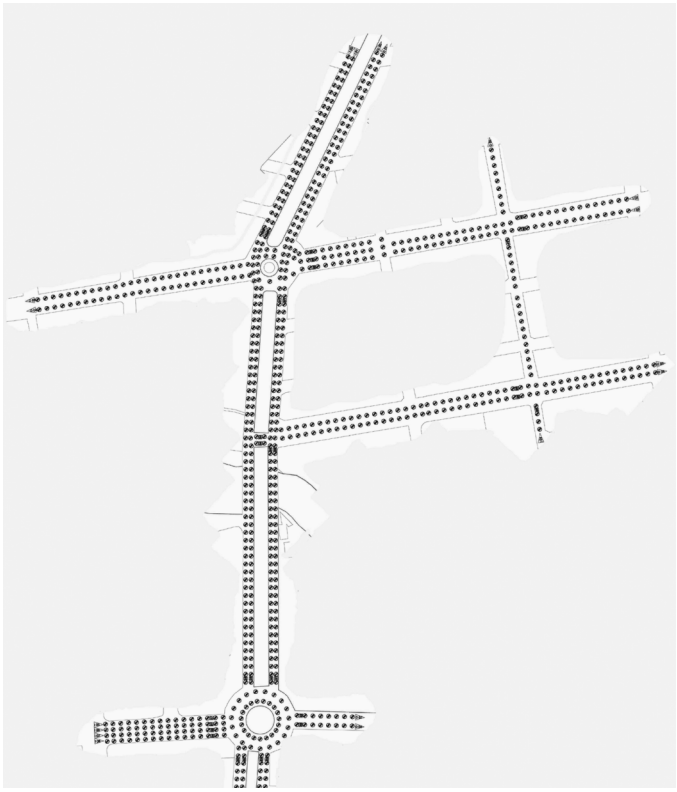


Fig. 18. Discretized traffic network—first half. The distance between two points is around 7 meters.

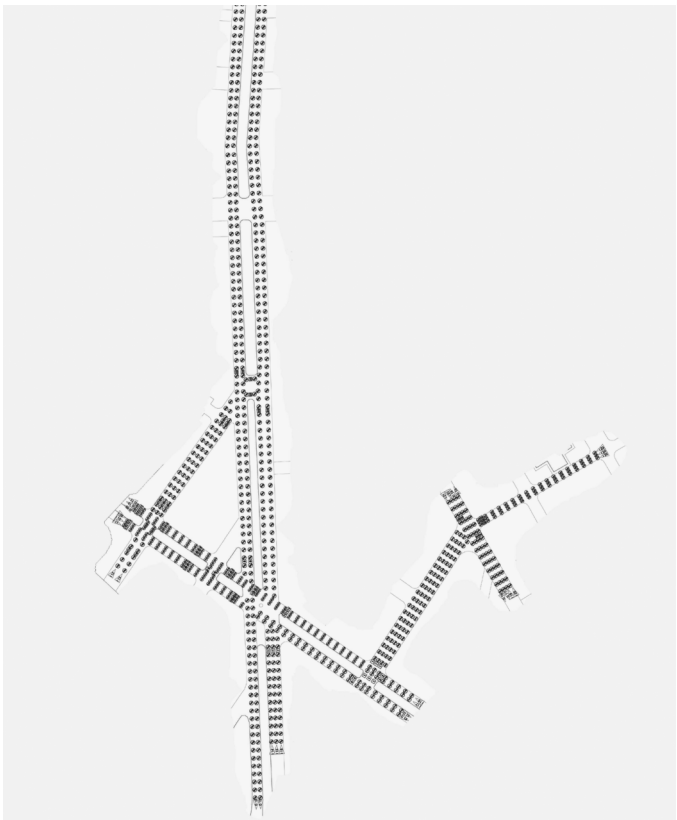


Fig. 19. Discretized traffic network—second half. The distance between two points is around 7 meters.

we have not considered those traffic lights controlling only pedestrians, i.e., for crossing a street.

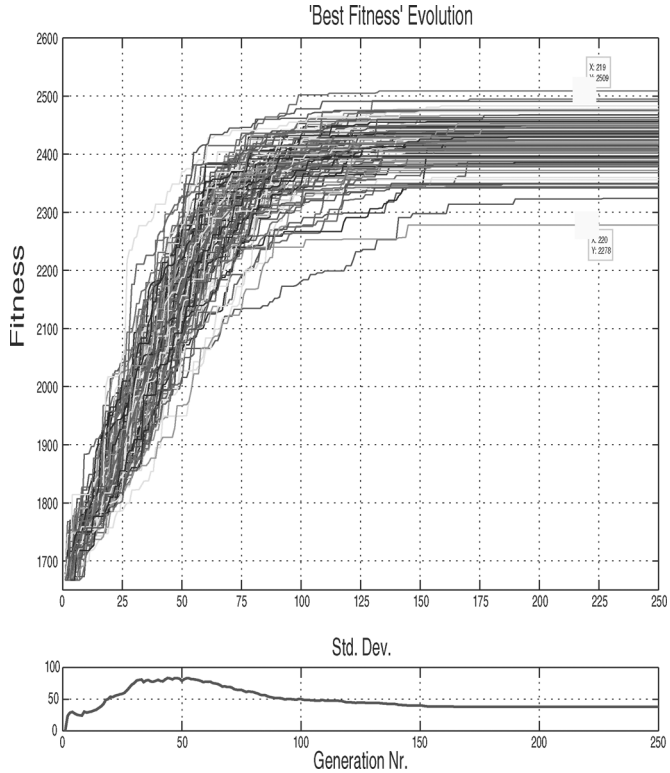


Fig. 20. Fitness evolution and standard deviation for the 50 optimizations performed.

With all this in mind and the discarded streets from Appendix I, we also have deprecated a set of traffic lights listed in Appendix II.

VII. Test Case Results

Throughout this section we describe the tests we carried out with the supplied data. These tests may be separated into two sets. First, we performed a set of optimizations searching for the best times for traffic light cycles. Then, we compared the traffic behavior using our results against the simulated behavior using the traffic departments supplied times.

In all, we ran 150 executions for this problem. For each of them we employed a 200 individual population within the GA, running through 250 generations. For each individual evaluation, we ran 2000 iterations of the microsimulation (2000 seconds). The mean execution time for each optimization is 4379.39 seconds (1 hour, 13 minutes, and 12 seconds). First, we present in Fig. 20 the fitness evolution for the 150 optimizations performed. In this figure, we plot the evolution of the fitness value of the best individual within each generation for the 150 solutions obtained. We represent the average "best fitness" plot with a thicker line. At the bottom of this figure, we represent the standard deviation of "best fitness" for the whole set of executions. In this figure, one may see that the algorithm converges clearly around 200 generations.

In Fig. 21, we represent the performance results using the solutions given to us by the City Council—the first nine points. The rest of the points represent the performance obtained using the solutions yielded by our method. One may observe that there

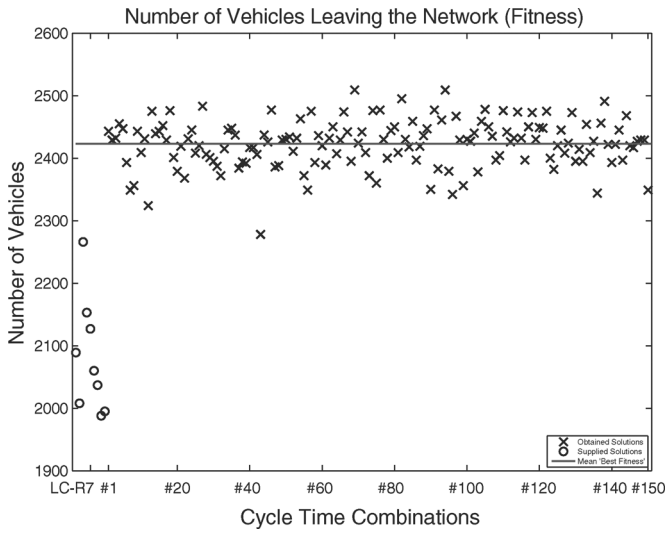


Fig. 21. Supplied versus obtained solutions.

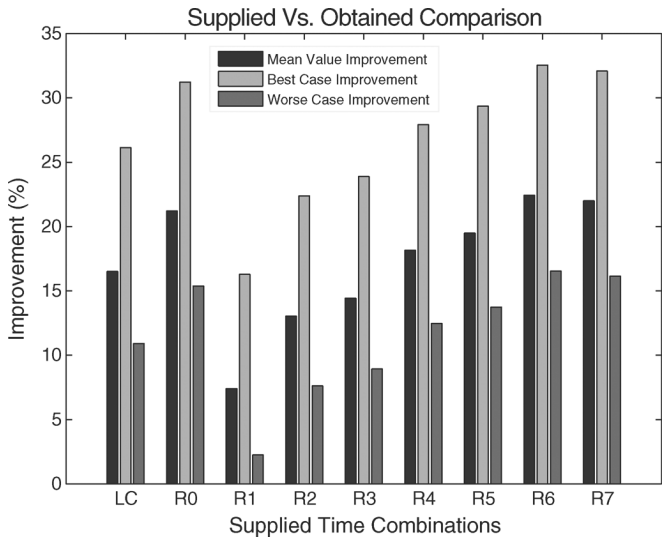


Fig. 22. Scaled (percent) comparison between supplied solutions versus ours.

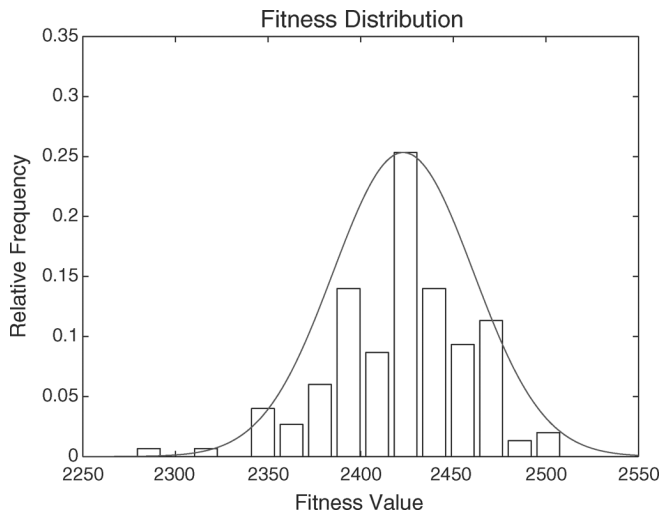


Fig. 23. Number of vehicles stochastic distribution versus a Gaussian probability distribution.

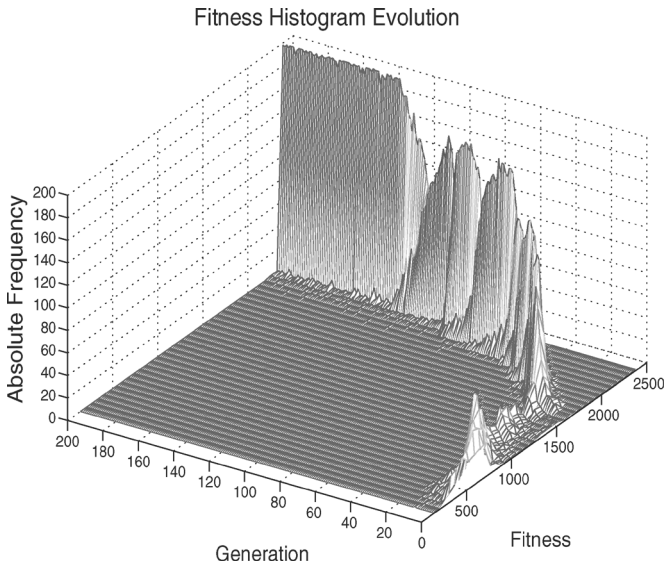


Fig. 24. Population fitness histogram evolution through the 200 generations.

is an obvious improvement using our times. Likewise, our 150 solutions seem to be more stable than theirs.

As a side note, one may observe that due to the stochastic nature of our optimization, solutions follow a *Gaussian* distribution, which may include extremal values (both high and low). In Fig. 23, we show the stochastic distribution of the best individual fitness value—the number of vehicles leaving the network—once each optimization is finished. We have also overlaid a *Gaussian* distribution with the same mean and standard deviation as in our results.

Fig. 22 shows the improvement—as a percentage—of the mean, best, and worst values of our 150 solutions against the nine supplied. This improvement stays within a range from 0.53 to 26.21. The smallest difference between the optimized results and the supplied simulated results is 12 vehicles—solution 43 with respect to supplied "R1." The biggest difference is 521 vehicles—distance from solution 69 to supplied "R6."

Finally, in Fig. 24, we represent the fitness distribution among the population as it evolved for one optimization execution. For each generation, we obtained a 35 interval histogram and we put this all together in this figure. Note that the population seems to converge in a high fitness zone—about 2400 vehicles leaving the network—as the optimization runs.

VIII. Conclusion and Future Work

One important conclusion is that we can clearly improve the supplied times in our simulated environment. So, we can seek optimal cycle time combinations for the traffic lights programming using our architecture with an appropriate amount of statistics. We have proven this with a real-world test case (nevertheless, using a simulated environment).

This is useful as reducing travel times in a city clearly means saving money and reducing environmental impact.

It is important to note that our system is intrinsically adaptable to particularized requirements, such as "Path" preferences, minimum, and maximum *stage lengths*, etc. In this sense, our system is flexible and adaptable.

In future work, we would like to use a more complete set of statistics which would help calculate a more accurate probability matrix.

In the next phase of our research, we will validate our traffic simulation using real-world data from another city. This is an important task we must accomplish, because, so far, we have relied on the results from previous studies involving traffic simulation using CA, but it is now time to do our own validation.

The next task will be a real challenge. We are looking for a city council that will allow us to test our results in a real traffic network, at least in a constrained manner. We realize that this is a very complicated objective. Once we have achieved the first experience, we think that city council officials in charge of traffic control will be less reluctant to collaborate.

Finally, we are considering the possibility of extending our model to take into account the "Pedestrians' Interaction" and including environmental aspects in the optimization criteria using a multiobjective approach.

APPENDIX I
STREETS IN/OUT OF OUR MODEL

A. Streets Taken Into Account

In this section, we enumerate the streets under consideration in this work.

- General Franco Boulevard
- General Mola Avenue
- Pulido Boulevard
- Asunción Avenue
- Ramón y Cajal Street
- San Sebastián Street
- Reyes Católicos Avenue
- Pablo Picasso Street
- Benito Pérez Armas Avenue
- Fragata Danmark Street
- Tomé Cano Street
- Juan Sebastián Elcano Street

B. Streets Withdrawn From Our Model

In this section, the streets withdrawn for simplifying purposes are listed.

- Heliodoro Rodríguez González Street
- Ángel Arocha Street
- Madrid Avenue
- Cineasta Miguel Brito Street
- Bernardino Seman Street
- Velázquez Street
- Unamuno Street
- Gilberto Cayol López Street
- Pintor Martín González Street
- Azorín Street
- Ramiro de Maeztu Street
- Eric Lionel Fox Street
- El Camello Street
- Legazpi Street
- del Carmen Avenue
- C. Sánchez Pinto Street
- General Goded Street
- G. Sanjurjo Street
- Calvo Sotelo Street
- Primo de Rivera Street
- 18 de Julio Street
- Castro Street
- Anselmo Street
- Álvarez de Lugo Street
- General Porlier Street
- General Serrano Street
- La Asunción Street
- Santiago Cuadrado Street
- Salamanca Street

APPENDIX II
TRAFFIC LIGHTS IN/OUT OF OUR MODEL

A. Traffic Lights Withdrawn From Our Model

In this section, we enumerate the traffic lights we have withdrawn from the model to simplify it (with reason).

- INTERSECTION #13
  - — Traffic light #2 (not needed).
  - — Traffic light #3 (not needed).
  - — Traffic light #5 (not needed).
  - — Traffic light #9 (not needed).
- INTERSECTION #16
  - — Traffic light #5 (we do not have its time chart).
- INTERSECTION #11
  - — Every traffic light [not enough average daily traffic (A.D.T.) data].
- INTERSECTION #15
  - — Every traffic light (not enough A.D.T. data).
- INTERSECTION #18
  - — Traffic light #3 (not enough A.D.T. data).
  - — Traffic light #9 (not enough A.D.T. data).
- INTERSECTION #21
  - — Every traffic light (not enough A.D.T. data).
- INTERSECTION #22
  - — Traffic light #5 (not enough A.D.T. data).
- INTERSECTION #23
  - — Every traffic light (not enough A.D.T. data).
- INTERSECTION #24
  - — Traffic light #9 (not needed).
- INTERSECTION #25
  - Traffic light #8 (not needed).
  - Traffic light #4 (not needed).
  - Traffic light #10 (not needed).
  - Traffic light #5 (not needed).
  - Traffic light #17 (not needed).
  - Traffic light #16 (not needed).
  - Traffic light #11 (not needed).
- INTERSECTION #27
  - — Every traffic light (not enough A.D.T. data).
- INTERSECTION #114
  - — Every traffic light (not enough A.D.T. data).
- INTERSECTION #117
  - — Every traffic light (not enough A.D.T. data).
- INTERSECTION #119
  - — Every traffic light (not enough A.D.T. data).

APPENDIX III
APPROXIMATIONS FOR THE ORIGIN-DESTINATION MATRIX

In this appendix, we present the approximations assumed for the origin-destination probability matrix calculation. It was necessary to take these approximations, since we did not have enough real data to produce a proper probability matrix. If we were given a whole set of street traffic statistics, no approximation would be needed.

- No traffic variation caused by streets excluded from the model is taken into account.
- INTERSECTION #25 (Plaza de la Paz):
  - The traffic from General Franco Boulevard is divided into three equal parts. Two of them go to Asunción Avenue and the other goes to General Mola Avenue.
  - The traffic coming from Pulido Boulevard is divided into three equal parts: General Franco, General Mola, and Asunción Avenue.
- INTERSECTION #22 (Plaza de Republica Dominicana):
  - We deprecate the traffic part that coming from Asunción Avenue, goes around Republica Dominicana Square and returns back to Asunción Avenue.
  - The traffic coming from Reyes Católicos Avenue is divided into two parts. The first part (10%) goes to San Sebastián Street and the rest (90%) heads to Asunción Avenue.
  - The traffic from Belgic Avenue is divided into three equal parts: Asunción Avenue, San Sebastián Street, and Reyes Católicos Avenue.
- INTERSECTION #20:
  - All the traffic moving through Reyes Católicos Avenue targeting at Republica Dominicana Square is composed of half coming from Pablo Picasso Street and the other half from Reyes Católicos Avenue.
  - The traffic coming from Pablo Picasso Street targeting at Reyes Católicos Avenue is divided into two parts. The first part (75%) goes to Republica Dominicana Square and the other part (25%) goes in the opposite direction.
- INTERSECTION #17:
  - All the traffic leaving this intersection towards the Reyes Católicos Avenue targeting at Republica Dominicana Square is composed of two equal parts. The first one coming from the TF-1 highway viaduct and the second one coming from Benito Pérez Armas Avenue.
  - The traffic entering Reyes Católicos Avenue from Benito Pérez Armas Avenue is divided into two parts. The first part (90%) goes towards the TF-1 highway exit and the other part (10%) goes on through Benito Pérez Armas Avenue.
- INTERSECTION #16:
  - The traffic entering Benito Pérez Armas Avenue from the TF-1 highway exit is divided into two parts. The first part (90%) goes on through Benito Pérez Armas Avenue and the other part (10%) goes towards Fragata Danmark Street.
  - The traffic entering Benito Pérez Armas Avenue from Fragata Danmark Street is divided into two parts. The first part (92.4%) goes towards Reyes Católicos Avenue and the other part (7.6%) goes in the opposite direction.

TABLE III
ORIGIN-DESTINATION PROBABILITY MATRIX EMPLOYED

Outputs/Inputs	0	5	10	12	18	19	21	24	26	27	29
1	-	33	18	19	16	19	8	12	9	5	9
3	33	33	18	8	6	8	3	5	3	2	3
4	-	1	2		-	-	-	-	-	-	-
11	11	5	45	6	5	6	2	4	2	2	3
14	10	5	3	33	1	1	1	1	1	1	1
16	-	-	-	-	-	3	4	9	4	2	7
21	-	-	-	-	5	-	-	-	50	33	-
23	41	20	13	29	10	30	5	11	5	4	62
26	-	-	-	-	5	-	50	-	-	33	-
28	5	2	2	4	53	33	27	59	27	18	15

- INTERSECTION #13:
  - The traffic coming from Juan Sebastián Elcano Street is divided into three equal parts. The first part goes towards Tomé Cano Street—targeting at Heliodoro Rodríguez López Street. The second part goes to Fragata Danmark, and the third part takes Tomé Cano Street in the direction towards 3 de Mayo Avenue.
  - The traffic going through Tomé Cano Street from Heliodoro Rodríguez López Street is divided in two halves, one towards Fragata Danmark and the other to Tomé Cano continuance street.
  - The traffic going through Tomé Cano Street directed to Heliodoro Rodríguez López Street is divided into two halves. One goes through Tomé Cano Street and the other entering Fragata Danmark Street.
- INTERSECTION #19:
  - The traffic coming by Picasso Street to the intersection by Benito Pérez Armas Avenue targeting at Reyes Católicos Avenue is divided into three equal parts. Two of them go in both directions of traffic in Benito Pérez Armas and the third one goes on through Picasso Street.
  - The traffic entering Picasso Street directed to Reyes Católicos Avenue is composed of three equal parts, coming from the two traffic directions of Benito Pérez Armas Avenue and Picasso Street.

A. Origin-Destination Probability Matrix for This Work

In this appendix, we are showing the Origin-Destination Probability Matrix employed (Table III). To generate it, we have used the average daily traffic (ADT)[12] data supplied with the help of the approximations listed in Appendix III.

APPENDIX IV
GRAY CODE CODIFICATION

The purpose of this appendix is to assist in the understanding of the Gray Code Codification we used.

In short, it works like this: First, we shift the binary number one bit to the right and prune the least significant bit (the last on the right). Then, we perform a binary XOR operation between the original number and the pruned one.

For example, the binary code of 5 is 101. If we do the binary XOR of 101 and 010 (the pruned number), we will get 111, since the three pairs of bits are different.

[12]ADT: The total volume during a given time period in whole days greater than one day and less than one year divided by the number of days in that time period.

TABLE IV
GRAY CODE CONVERSION EXAMPLES

Integer	Binary	Pruned Gray	
0	000	000	000
1	001	000	001
2	010	001	011
3	011	001	010
4	100	010	110
5	101	010	111
6	110	011	101
7	111	011	100

In Table IV, we show more examples including the pruned number.

To do the gray to binary conversion an iterative routine is used. The most significant bit (MSB) is copied from the Gray Coded number to the binary number. The other bits are calculated like this: If the Gray Coded number bit is "1," we put the complementary of the last written bit. If the gray bit is "0," we repeat the last written bit value.

For example, 4 is 110 in Gray Code. The MSB of the corresponding binary number will be the MSB of the gray number ("1"). The second bit of the Gray Coded number is "1," so we have to write the 2's complement of the last written bit ("0"). Finally, the third bit of the gray number is "0." So we will repeat "0." The resulting binary number is 100.

ACKNOWLEDGMENT

The authors would like to acknowledge the Santa Cruz de Tenerife City Council Traffic Department for their kind help. They specially would like to thank Mr. H. Rodríguez González for his willingness to collaborate with them, making this work feasible. They also would like to acknowledge Dr. D. Shea from the University of Las Palmas de Gran Canaria for his kind assistance copy editing the original manuscript.

REFERENCES

[1] E. Brockfeld, R. Barlovic, A. Schadschneider, and M. Schreckenberg, "Optimizing traffic lights in a cellular automaton model for city traffic," 2001. [Online]. Available: http://www.arxiv.org/ps/cond-mat/0107056

[2] N. Rouphail, B. Park, and J. Sacks, "Direct signal timing optimization: Strategy development and results," in *Proc. XI Pan Amer. Conf. Traffic and Transportation Eng.*, Gramado, Brazil, 2000. [Online]. Available: http://citeseer.ist.psu.edu/383145.html

[3] A. Vogel, C. Goerick, and W. von Seelen, "Evolutionary algorithms for optimizing traffic signal operation," in *Proce. ESIT 2000*, Aachen, Germany, 2000, pp. 83–91.

[4] S. López, P. Hernandez, A. Hernandez, and M. Garcia, "Artificial neural networks as useful tools for the optimization of the relative offset between two consecutive sets of traffic lights," in *Foundations and Tools for Neural Modeling*. Berlin, Germany: Springer-Verlag, 1999, LNCS, pp. 795–804.

[5] A. Halati, H. Lieu, and S. Walker, "CORSIM—corridor traffic simulation model," in *Proc. 76th Ann. Meeting Transp. Res. Board*, Washington, D.C., 1997, pp. 570–576.

[6] O. Tveit, "Common cycle time—A strength or barrier in traffic light signaling," *Traffic Eng. Control Mag.*, vol. 44, no. 1, pp. 19–21, 2003.

[7] G. Y. Lim, J. J. Kang, and Y. S. Hong, "The optimization of traffic signal light using artificial intelligence," in *Proc. 10th IEEE Int. Conf. Fuzzy Syst.*, Dec. 2–5, 2001, vol. 3, pp. 1279–1282.

[8] J. Sánchez, M. Galán, and E. Rubio, "Genetic algorithms and cellular automata: A new architecture for traffic light cycles optimization," in *Proc. Congr. Evol. Comput.*, 2004, vol. II, pp. 1668–1674.

[9] A. Di Febbraro, D. Giglio, and N. Sacco, "On applying Petri nets to determine optimal offsets for coordinated traffic light timings," in *Proc. IEEE 5th Int. Conf. Intell. Transportation Syst.*, 2002, pp. 773–778.

[10] L. Li, N. Tang, X. Mu, and F. Shi, "Implementation of traffic lights control based on petri nets," *IEEE Intell. Transp. Syst.*, vol. 3, pp. 1749–1752, 2003.

[11] M. J. Smith, "Optimum network control using traffic signals," in *Proc. IEE Colloquium on UK Developments in Road Traffic Signalling,*, May 5, 1988, pp. 8/1–8/3.

[12] Y.-S. Hong, J. S. Kim, J. Kwangson, and C. K. Park, "Estimation of optimal green time simulation using fuzzy neural network," in *Proc. Int. Conf. Fuzzy Syst.*, 1999, vol. 2, pp. 761–766.

[13] J. C. Spall and D. C. Chin, "A model-free approach to optimal signal light timing for system-wide traffic control," in *Proc. IEEE 33rd Conf. Decision Control*, 1994, vol. 2, pp. 1868–1875.

[14] Y.-S. Hong, J. Hyunsoo, and P. Chong-Kug, "New electrosensitive traffic light using fuzzy neural network," *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 6, pp. 759–767, 1999.

[15] M. Wiering, J. Vreeken, J. van Veenen, and A. Koopman, "Simulation and optimization of traffic in a city," in *Proc. IEEE Intell. Vehicles Symp.*, 2004, pp. 453–458.

[16] P. E. Black, *"Gray Code," From Dictionary of Algorithms and Data Structures*, P. E. Black, Ed. Gaithersburg, MD: NIST. [Online]. Available: http://www.nist.gov/dads/HTML/graycode.html

[17] J. Sánchez, M. Galán, and E. Rubio, "Bit level versus gene level crossover in a traffic modeling environment," in *Proc. Int. Conf. Comput. Intell. Modelling Control Automa. and Int. Conf. Intell. Agents, Web Technol. Internet Commerce*, 2005, vol. 1, pp. 1190–1195, ISBM: 0–7695–2504–0–01.

[18] M. Matsumoto and T. Nishimura, "Mersenne twister: A 623-dimensionally equidistributed uniform pseudorandom number generator," *ACM Trans. Modeling Computer Simulation*, vol. 8, no. 1, pp. 3–30, 1998.

[19] D. Helbing, "An improved fluid dynamical model for vehicular traffic," *Physical Rev. E*, vol. 51, p. 3164, 1995.

[20] B. S. Kerner and P. Konhäuser, "Structure and Parameters of Clusters in Traffic Flow," *Physical Review E*, vol. 50, no. 1, pp. 54–83, Jul. 1994, American Physical Society.

[21] R. D. Kühne and M. B. Rödiger , "Macroscopic simulation model for freeway traffic with jams and stop-start waves," in *Proc. 23rd Conf. Winter Simulation*, 1991, pp. 762–770, ISBN: 0–7803–0181–1, IEEE Computer Society.

[22] R. D. Kühne, "Verkehrsablauf auf Fernstrassen," *Phys. Bl.*, vol. 47/3, no. 201, pp. 201–204, 1991.

[23] H. J. Payne, "Freflo: A macroscopic simulation model of freeway traffic," *Transp. Res. Record*, vol. 722, no. 68, pp. 68–77, 1979.

[24] G. B. Whitham, *Linear and Nonlinear Waves*. New York: Wiley, 1974.

[25] C. F. Daganzo, "Requiem for second order fluid approximations of traffic flow," *Transp. Res. B*, pp. 277–286, 1995.

[26] S. Benjaafar, K. Dooley, and W. Setyawan, Cellular automata for traffic flow modeling Intelligent Transportation Systems Institute, Rep. No. CTS 97-09, 1997.

[27] K. Nagel and A. Schleicher, "Microscopic traffic modeling on parallel high performance computers," *Parallel Comput.* vol. 20, pp. 125–146, 1994. [Online]. Available: http://www.citeseer.ist.psu.edu/nagel93microscopic.html

[28] M. Cremer and J. Ludwig, "A fast simulation model for traffic flow on the basis of Boolean operations," *Math. Comput. Simulation*, vol. 28, pp. 297–303, 1986.

[29] J. von Neumann, "The general and logical theory of automata," in *John von Neumann—Collected Works*, A. H. Taub, Ed. New York: Macmillan, 1963, vol. V, pp. 288–328.

[30] E. Brockfeld, R. D. Khne, and P. Wagner, "Towards benchmarking microscopic traffic flow models networks for mobility," in *Proc. Int. Symp.*, 2002, vol. I, pp. 321–331.

[31] S. Krauss, P. Wagner, and C. Gawron, "Metastable states in a microscopic model of traffic flow," *Phys. Rev. E*, vol. 55, pp. 5597–5605, 1997.

[32] A. Schadschneider, D. Chowdhury, E. Brockfeld, K. Klauck, L. Santen, and J. Zittartz, "A new cellular automata model for city traffic," in *Traffic and Granular Flow '99: Social, Traffic, and Granular Dynamics*. Berlin, Germany: Springer-Verlag, 1999.

[33] K. Nagel and M. Schreckenberg, "A cellular automaton model for freeway traffic," *J. de Physique I France*, vol. 33, no. 2, pp. 2221–2229, 1992.

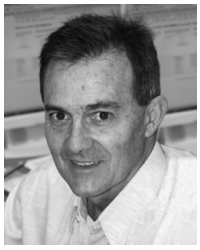
[34] O. Biham, A. A. Middleton, and D. Levine, *Phys. Rev. A*, vol. 46, p. 6124, 1992.
[35] J. Sánchez, M. Galán, and E. Rubio, "Stochastic vs deterministic traffic simulator. Comparative study for its use within a traffic light cycles optimization architecture," in *Proc. Int. Work Conf. Interplay Between Natural Artif. Comput.*, 2005, vol. II, pp. 622–631.

Javier Sánchez received the B.S. degree in 1998 and the M.S. degree in telecommunications engineering from the University of Las Palmas de Gran Canaria, Las Palmas, Spain, in 2001. Since 2002, he has been working towards the Ph.D. degree at the Innovation Center for Information Society (C.I.C.E.I.), University of Las Palmas de Gran Canaria.

During this time, he has collaborated in several research projects. In 2003, he was given a research grant by the Canary Islands Government.

Manuel Galán received the Math degree in algebra and geometry from the University of Valladolid, Valladolid, Spain, in 1985, and the Ph.D. degree in mathematics from the University of Las Palmas de Gran Canaria, Las Palmas, Spain, in 1994.

He has been a Professor of Mathematics since 1994. Currently, he is a Professor at the School of Architecture, University of Las Palmas de Gran Canaria. He has gained considerable experience in systems resolution and genetic algorithms. He is coeditor and coauthor of the book *Genetic Algorithms in Engineering and Computer Science* (Wiley, 1996).



Enrique Rubio received the undergraduate degree in physics from the University Complutense of Madrid, Madrid, Spain, in 1971, and the Ph.D. degree from the University of La Laguna, La Laguna, Spain, in 1980.

He has been a Professor since 1990. He is also the Director of the Innovation Center for Information Society (Research Center), University of Las Palmas de Gran Canaria, Las Palmas, Spain. He has considerable experience in information society and technology.