2017 16th IEEE International Conference on Machine Learning and Applications

Evolving Adaptive Traffic Signal Controllers for a Real Scenario Using Genetic Programming with an Epigenetic Mechanism

Esteban Ricalde
Department of Computer Science
Memorial University of Newfoundland
St. John's, NL, A1B 3X5, Canada
Email: ergl45@mun.ca

Wolfgang Banzhaf
Department of Computer Science
Memorial University of Newfoundland
St. John's, NL, A1B 3X5, Canada
Email: banzhaf@cs.mun.ca

***Abstract*—An important challenge for traffic signal control is adapting to irregular changes in traffic. In recent years, different heuristics have been developed to address this issue. However, most of them are tested in artificial scenarios under controlled circumstances.**

In this paper, we present the first implementation of Genetic Programming in the evolution of traffic signal controllers for a real-world scenario. The evolved controllers are compared with a static control and an actuated control. The results indicate a significant improvement over traditional methods. Moreover, additional experiments indicate that the evolved controllers have the ability to adapt to unplanned changes in traffic conditions.

***Index Terms*—Real-world application, Traffic signal control, Machine learning, Genetic programming**

I. Introduction

Urban traffic signal control is a complex nonlinear problem and traffic congestion affects daily life of many citizens. The rapid increase of metropolitan populations makes control of traffic signals a challenging task.

The optimization of time schedules is one of the traditional solutions to this problem. This operation can be performed through expert observation or through heuristic methods [1], [2], [3]. However, given that traffic congestion is a dynamic phenomenon, this approach has the downside of requiring constant updating of the schedules. Furthermore, it requires constant monitoring and design of contingency plans to react to atypical congestion events.

The use of actuated controls and vehicle detectors allows the generation of more adaptable solutions. Different methods have been implemented to develop efficient, actuated traffic signal rules or controllers. Some of them are domain specific. For example, Green-wave method [4] and back-pressure [5]. Others are based in group dynamics. For example, self-organized systems [6] and auctions [7].

However, these solutions are heuristics specially designed by humans to solve the traffic signal control problem. The present work explores the idea of using machine learning to *automatically* evolve adaptive controllers for a real-world traffic scenario. We are trying to answer the question: Is a computer able to generate a better solution than a heuristic designed by humans for a complex traffic network with multiple intersections?

The remaining of the paper is organized as follows: Section II lists recently published solutions to the Traffic Signal Control Problem, Section III presents the characteristics of the traffic simulator and traffic scenario used, Section IV briefly defines Genetic Programming and how it can be used to evolve traffic controllers, Section V introduces the proposed modification to the standard Genetic Programming, Section VI presents the experiments performed and the results obtained, Section VII summarizes the discussion, and Section VIII draws the conclusions.

II. Previous work

Different approaches have been used to generate actuated traffic controllers. This section presents some of the recent publications related to the topic.

In [8], Braum and Kemper modified BALANCE [9]. They replaced the hill-climbing algorithm used at the tactical level of BALANCE with a Genetic Algorithm. Several experiments were performed with the traffic network of Ingolstadt, Germany. The results demonstrate a better performance over the original BALANCE. The system was implemented in the real-world and daily average delays were reduced by 21%.

Padmasiri and Ranasinghe [10] used a Genetic Programming and fuzzy logic hybrid approach to define fuzzy rules for a scenario with a single intersection under different traffic densities. The set of evolved rules uses traffic parameters as input and decides to extend or terminate the current green lapse. Even though the results present an improvement over previous work, the solutions lack adaptability to changes under traffic congestion and the study only considers a single intersection.

Zubillaga et al. [6] proposed a set of self-organizing rules to coordinate urban traffic. They compared their method with the green wave method in a 10×10 grid network under different traffic densities. The average velocity of the vehicles traversing the network is higher when the self-organized n

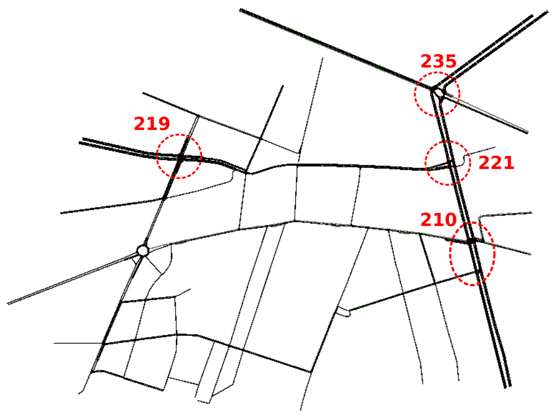

Fig. 1. Network to be optimized.

TABLE I
SUMMARY OF THE CONFIGURATION PARAMETERS FOR GP

Configuration Parameters	**Selected Values**
Population size	40 individuals
Number of generations	200
Fitness function	Average vehicle delay
Function set	condition, addition, subtraction, conjunction, disjunction, equal to and bigger than
Terminal set	vQueue, hQueue, integer values between -5 and 5
Mutation probability rate	20% per tree
Crossover probability	80%
Initial size limit	5 levels
Maximum size limit	7 levels
Selection operator	Tournament selection with group size of 7 individuals
Elitism	1 individual

In [7], an auction-based method is proposed to coordinate phase switching operations using local induction loop information. The method was tested in a scenario generated with real-world traffic data from the Mountain View, California area. Auction-based control performs better than static lights and a planning-based method.

Yuan et al. [11] proposed a dynamic slot time mechanism to control the back-pressure algorithm [5]. The mechanism is tested in a 4×4 artificial network with a specific traffic density. Partially successful results are reported over the original back-pressure algorithm.

Yang and Ding [4] proposed a breadth-first-search approach to the green wave method using gap-outs and extensions. The method was compared with a self-organized algorithm in a traffic network of part of Qidong, China. The green wave method works better than the self-organized method when traffic saturation is high.

In [12], we presented an epigenetic modification of Genetic Programming to solve a traffic signal control problem in a small artificial scenario. The results show an increment in the performance compared to other methods working with a basic simulator.

III. Simulator

We use SUMO [13] for all the simulations reported in this paper. SUMO (Simulation of Urban MObility) is an open-source traffic micro-simulation suite that has been available since 2001. We selected SUMO because it includes a traffic control interface called TraCI. It allows the retrieval of values of simulated objects and on-line closed loop feedback [14] using a Python interface.

A. Traffic Scenario

Most of the research projects on heuristic methods in traffic signal problems work with basic scenarios. Most of these scenarios are artificial and distant from real-world traffic conditions. Recently, different scenarios of real cities have been released for SUMO suite to be used by the research community [15], [16]. These scenarios have the correct representation of real networks and include real-world traffic data.

We decided to test our algorithm in one of them. Due to the evaluation time required by our method, we chose the Andrea Costa scenario of Bologna city in Italy [16] presented in the Figure 1. The model represents 2.45 km^2 of a real city with a total of 179 edges, 112 nodes, and seven traffic lights.

The scenario includes one hour of real traffic for the morning peak between 8:00 am and 9:00 am. More than 8600 private vehicles and 160 buses are included in the scenario. As described in [16], the city of Bologna uses the UTOPIA system for traffic light control. UTOPIA [17] optimizes traffic light schedules and sorts the traffic light phases to satisfy traffic demand. For this paper, all the experiments are executed using the traffic schedule generated by UTOPIA.

A preliminary analysis of the scenario allowed us to identify the intersections more affected by traffic congestions. We focused our development on controllers for the four traffic lights circled in Figure 1 because they are the most congested intersections.

B. Actuated control

Our approach is compared with an actuated control already implemented in the SUMO suite. The method is based on Time Gaps [18]. It works by prolonging traffic phases whenever a continuous stream of traffic is detected. The actuated control affects the cycle duration in response to dynamic traffic conditions.

For the actuated control of the four intersections, we used the maximum and minimum duration parameters generated by UTOPIA and included in the Andrea Costa scenario.

IV. Genetic Programming

Genetic Programming (GP) [19] is an Evolutionary Algorithm where computer programs are encoded as a set of genes and evolve to perform well on a specific task. Usually, GP generates an executable program as output. The GP parameters are presented in Table I. The parameters are similar to those used in [12].

This paper uses strongly typed GP [20] and a tree-based representation similar to the one used in [12] to simultaneously

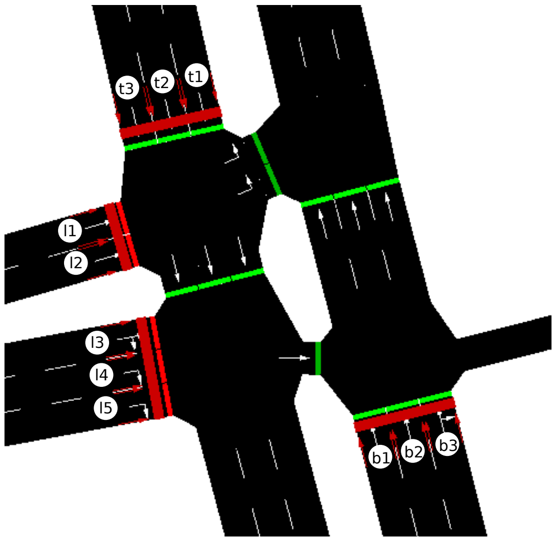

Fig. 2. Detail of intersection 221 with detectors.

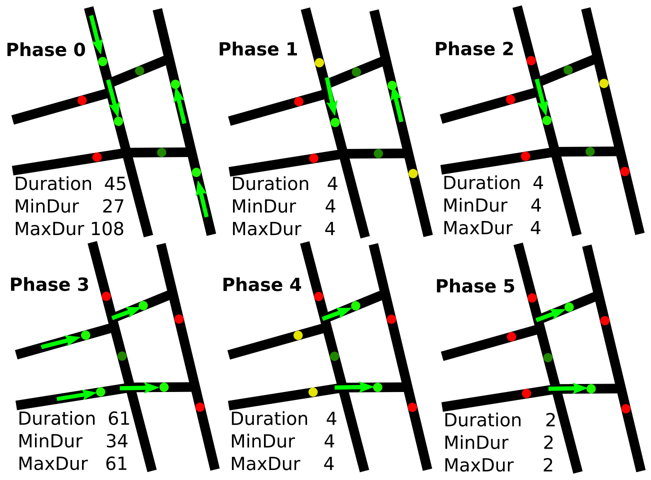

Fig. 3. Time schedule generated by UTOPIA for intersection 221.

evolve the four traffic signal controllers for each of the intersections circled in Figure 1.

A. Problem configuration

In the optimization of traffic lights, each intersection represents a different problem. This is because the topology, traffic densities and traffic time schedules are different for each intersection. Therefore, instead of evolving a general controller, we evolve a controller for each of the intersections to be optimized.

Each of the controllers is a different implementation of the `trafficRule` Python function, where `vQueue` is the sum of the vehicles stopped in the north-south direction and the vehicles stopped in the south-north and `hQueue` is the sum of the vehicles stopped in the west-east direction and the vehicles stopped in the east-west direction.

```
def trafficRule(vQueue,hQueue):
    if vQueue > hQueue + 4:
        return 1
    else:
        if hQueue > vQueue + 4:
            return -1
    return 0
```

Each controller is executed twice for the corresponding intersection cycle. Two or more phases are selected to be affected by the controller. However, the phases can only be incremented or decremented in the range defined by the minimum duration and maximum duration of the specific phase. This requires the manual analysis of each intersection to be optimized and relates the detectors of the different traffic directions to the variables `vQueue` and `hQueue`.

As an example, Figure 2 presents the intersection 221 with all its detectors. Figure 3 presents the time schedule generated by UTOPIA for the same intersection. In this case, the association is simple: `vQueue` is equal to the sum of detectors t1, t2, t3, b1, b2 and b3. Meanwhile, `hQueue` is equal to the sum of detectors l1, l2, l3, l4 and l5. The value returned by the function is added to the duration of phase 0 and subtracted from phase 3. However, the association of detectors and selection of affected phases for intersections 210 and 235 required a deeper analysis.

During the GP evaluation phase, the Python translator generates the corresponding controllers. After these programs are generated, the simulator is executed with the parameters of the Andrea Costa scenario. In order to have access and control over traffic signals, an additional communication layer was implemented between the evolutionary process and the simulation through the TraCI Interface. An architecture diagram of the modifications implemented is displayed in Figure 4.

It is important to emphasize that the framework generated is open-source and only minor modifications are required in the experiment runner to test other machine learning methods with the same scenario[1].

V. Epigenetics in Evolutionary Algorithms

In Biology, Epigenetics can be defined as the study of cellular and physiological phenotypic trait variations that are caused by external or environmental factors affecting the way cells read, express and transfer genetic material.

In recent years, the Evolutionary Algorithm (EA) community has been focusing in the discoveries in the area of Epigenetics. Different approaches ([21], [22], [23]) have been used to represent phenotypic mechanisms. Epigenetics has normally been used in EA as an additional optimization step to accelerate the adaptation of the method.

In [12], we presented an epigenetic modification of Genetic Programming to solve a traffic signal control problem in an small artificial scenario. The modification is based in DNA methylation. It changes the chromosome activation when the

[1]Code available in https://sourceforge.net/projects/acostatrafficscenariotester/

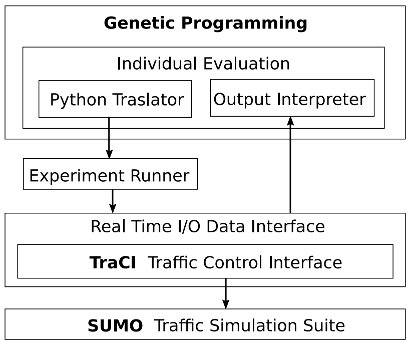

Fig. 4. Integration of SUMO in the evolution process.

traffic conditions are modified. The results show an increment in the performance compared to other methods.

A. Epigenetic mechanism

This paper extends the method proposed in [12] to be able to work with the SUMO simulator in a real world traffic network. For terms of practicality, we will identify as EpiGP when Genetic Programming with the epigenetic mechanism is used. For a more detailed description of the epigenetic mechanism please refer to [12].

An activation index is associated with every conditional node in the chromosome. All the activation indices of an individual are stored in an epigenetic vector for easy manipulation. In the first generation, the epigenetic vector is randomly initialized with values between 0% and 100%. During the individual evaluation step, any conditional node with an activation index smaller than the activation threshold (50% for this experiment) is ignored. An online adaptive process, described in detail in [12], is used to modify the epigenetic vector of the individual. After the evaluation, The vector is transferred to the offspring as part of the crossover operation.

The Python class `Controller0` is an example of the output generated by the Python translator including the epigenetic vector. In this case, the second condition is ignored because the activation index is smaller than the activation threshold. In other words, `trafficRule` will return 0 even if `hQueue > vQueue + 4` is true.

```
class Controller0(controller.Controller):
    def trafficRule(vQueue,hQueue):
        if (self.epiVect[0] > 0.5 and
            vQueue > hQueue + 4) :
                return 1
        else:
            if (self.epiVect[1] > 0.5 and
                hQueue > vQueue + 4):
                    return -1
        return 0

    def __init__(self):
        self.epiVect = [0.95, 0.4]
```

With this approach, GP can evolve controllers with different behavior under traffic congestion conditions and in stable conditions. The mechanism works as a local hill-climber to adapt the intersection behavior to the environmental changes.

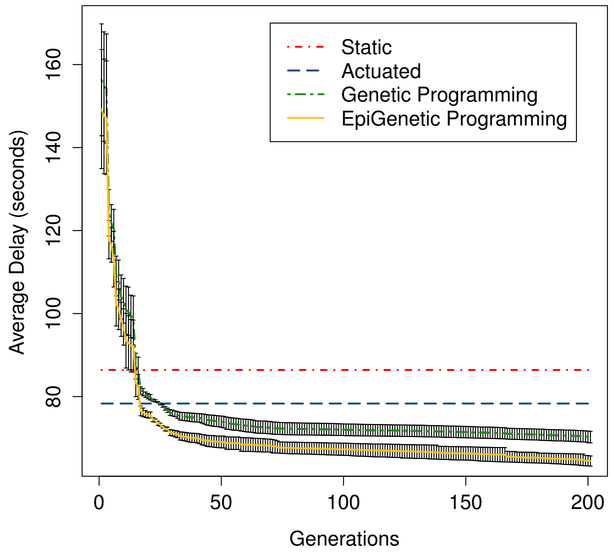

Fig. 5. Average delay comparison of the three methods for the Andrea Costa scenario using the real traffic peak configuration.

VI. Experiments

Four different methods for traffic signal control were tested in the Andrea Costa scenario: (1) static control; (2) the actuated control based on time gaps described in section III-B; (3) the evolution of adaptive controllers using GP, described in subsection IV-A; and (4) the evolution of adaptive controllers using EpiGP, described in subsection V-A. All the methods use the UTOPIA time schedules included in the Andrea Costa scenario.

For the time gap actuated method, GP and EpiGP, only the intersections 210, 219, 221 and 235 indicated in Figure 1 are being optimized.

The average delay is used as the fitness function. Average delay is defined as the sum of the stopped time of all the vehicles in the system divided by the number of processed vehicles. The average delay is commonly used to measure the performance of traffic signal control methods [8], [4].

The same route files are used for each simulation run. Therefore, the output for each execution of the static and actuated methods are the same. Hence, only one run was required for the fixed static control and the actuated control. However, five independent runs were performed for GP and EpiGP.

Figure 5 presents the comparison of the results obtained by the four methods. For GP and EpiGP the average and standard error of the best individual of the five runs are displayed for each generation.

As expected with a randomly generated initial population, during the first 10 generations of GP and EpiGP, the values are even worse than the value of the static control. However, EpiGP only requires an average of 25 generations to evolve controllers that outperform the average delay produced by the actuated control. After that, the learning curve decelerates, but it keeps improving the solutions. The learning remains constant until the end of the evolution. Therefore, an optimal set of controllers has not been found yet after 200 generations.

The epigenetic modification helps GP to find better solutions. Figure 5 presents a similar behavior for GP and EpiGP during the first 30 generations. However, the experiments with the epigenetic mechanism are able to generate solutions that perform better in the scenario in later stages of the evolution process.

A. *Different Traffic Densities*

Any machine learning method has the risk of over-training solutions. In other words, they can generate solutions only suitable for the data frame used during the training phase. An over-trained solution lacks the capacity to perform well even if the data frame is slightly modified. GP is not an exception to this phenomenon.

In our experiment, over-trained signal traffic controllers would not be able to adapt to traffic conditions different to the hour used as the training set. This would mean that different controllers would be required throughout the day. However, the idea of evolving controllers is to eliminate the requirement of scheduled modifications. To consider the experiment a success, the solutions generated by EpiGP should perform better than the static control and the actuated control under different traffic densities.

An additional experiment with different traffic densities was realized to guarantee that the solution generated by the evolutionary methods in the previous experiment were not over-trained with the traffic conditions used in the Andrea Costa scenario. Ideally, the controllers would be tested with real data of different hours for the same network. However, the Andrea Costa scenario only provides traffic data for a single hour. To overcome this issue, three new scenarios with different traffic densities and random routes where generated for the Andrea Costa network using the `randomTrips.py` tool provided with the SUMO suite [18]. Each scenario has a length of 60 minutes.

Figure 6 presents the comparison of the four different methods for the three different density scenarios. The average and standard error of the five independent experiments is presented for the standard GP method and for EpiGP.

Table II presents the percentage of improvement of the actuated control, GP and EpiGP over the static control for the four different experiments realized in the current paper. Negative values indicate that the static method generated a better solution than the given method. In three of the four experiments, EpiGP outperformed the other methods. However, the level of improvement is less than the obtained for the experiment using real data, as it can be seen in the second, third and fourth columns of Table II.

TABLE II
AVERAGE DELAY RELATIVE DIFFERENCES WITH STATIC CONTROL

Methods	Training	Low Density	Medium Density	High Density
Actuated	9.36 %	**20.1 %**	-15.17 %	-41.18 %
GP	18.70 %	1.42 %	3.50 %	9.09 %
EpiGP	**23.23 %**	6.33 %	**6.93 %**	**12.13 %**

The behavior of the actuated method in these scenarios presents some particularities. It obtained the best performance in the scenario with low density. However, it behaved poorly in the scenarios with medium and high densities. The actuated method works by prolonging traffic phases whenever a continuous stream of traffic is detected. However, especially in the scenario with high traffic density, several of the intersections are constantly saturated. Therefore, the actuated method is not able to optimize the duration of the phases when traffic is saturated in all the directions.

The evidence presented demonstrates that the signal traffic controllers generated by both evolutionary methods are not overtrained for the hour of traffic peak used during the evolution process. Consequently, the controllers can be used throughout the day and reduce the average delay of the vehicles circulating in this specific section of the city.

VII. DISCUSSION

The experiments determined that Genetic Programming is able to generate traffic signal controllers adaptable to traffic variation for a real scenario of small size. The method was compared with a static control and an actuated control. The generated solutions were tested using scenarios with different traffic densities. The results display improvement over an actuated control included in SUMO suite based on time gaps.

Genetic Programming is evolving an independent controller for each intersection. This approach differs from other heuristics where the goal is a general control mechanism. Hence, the adaptive local controllers generated by the method are able to adapt to unexpected traffic density changes.

The adaptive controllers generated do not require communication between lights, and modifications to the traffic schedules are based on local information. The EpiGP and GP use information provided only through local induction-loop sensors to generate reactive signal controllers. This would reduce the cost of its implementation in the real-world. However, the model can be easily extended to consider communication between lights or between cars and lights.

VIII. CONCLUSIONS AND FUTURE WORK

In [12], we introduced a epigenetic variation of GP to optimize traffic signal controllers using synthetic data. The current paper presents a proof of concept of a similar algorithm using a real-world scenario. The future work should direct towards the optimization of larger realistic networks and tweaking the method to be efficient under different conditions.

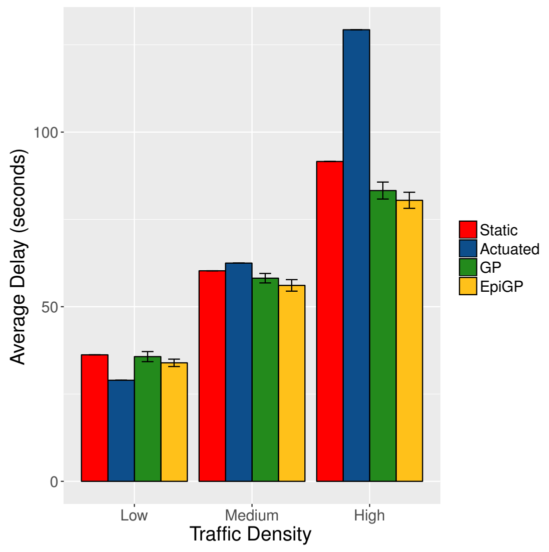

Fig. 6. Comparison of scenarios with different densities.

All experiments presented in this paper used the traffic schedule generated by UTOPIA. Additional experiments are required to analyze the performance of EpiGP when an optimized traffic schedule is not available.

Larger scenarios should be evaluated to analyze the behavior of the method under different circumstances. Codeca, et al. [15] present a full city scenario with 24 hours of real traffic data. However, testing EpiGP in a scenario of such size requires modifications to the architecture proposed in the current paper.

The computational time required for execution of GP constrained the number of independent runs made. An alternative solution is to use parallel computing to reduce simulation clock time. However, a different traffic simulator needs to be used for this task because SUMO does not allow parallel computation.

This method can be compared with a broader set of methods used to optimize traffic signal control. Examples are the self-organizing method described in [6] and the green wave method described in [4].

Acknowledgments

We want to thank the time and help provided by Dario Zubillaga and Carlos Gershenson regarding SUMO and TraCI technical details.

The first author gratefully acknowledges the financial support provided by Memorial University of Newfoundland.

References

[1] J. J. Sánchez-Medina, M. J. Galán-Moreno, and E. Rubio-Royo, “Traffic signal optimization in la almozara district in saragossa under congestion conditions, using genetic algorithms, traffic microsimulation, and cluster computing,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 1, pp. 132–141, 2010.

[2] X. Nie, Y. Li, and X. Wei, “Based on evolutionary algorithm and cellular automata combined traffic signal control,” in *3rd International Symposium on Knowledge Acquisition and Modeling (KAM)*. IEEE, 2010, pp. 285–288.

[3] M. B. Younes and A. Boukerche, “Intelligent traffic light controlling algorithms using vehicular networks,” *IEEE Transactions on Vehicular Technology*, vol. 65, no. 8, pp. 5887–5899, 2016.

[4] Z. Yang and Z. Ding, “Actuated green wave control for grid-like network traffic signal coordination,” in *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2016, pp. 000 953–000 958.

[5] T. Wongpiromsarn, T. Uthaicharoenpong, Y. Wang, E. Frazzoli, and D. Wang, “Distributed traffic signal control for maximum network throughput,” in *15th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2012, pp. 588–595.

[6] D. Zubillaga, G. Cruz, L. D. Aguilar, J. Zapotécatl, N. Fernández, J. Aguilar, D. A. Rosenblueth, and C. Gershenson, “Measuring the complexity of self-organizing traffic lights,” *Entropy*, vol. 16, no. 5, pp. 2384–2407, 2014.

[7] M. Covell, S. Baluja, and R. Sukthankar, “Micro-auction-based traffic-light control: Responsive, local decision making,” in *IEEE 18th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2015, pp. 558–565.

[8] R. Braun and C. Kemper, “An evolutionary algorithm for network-wide real-time optimization of traffic signal control,” in *IEEE Forum on Integrated and Sustainable Transportation System (FISTS)*. IEEE, 2011, pp. 207–214.

[9] B. Friedrich, “Balance and control: Methods for traffic adaptive control,” in *World Congress on Intelligent Transport Systems (2nd: 1995: Yokohama-shi, Japan). Steps forward Vol. 5*, 1995.

[10] T. Padmasiri and D. Ranasinghe, “Genetic programming tuned fuzzy controlled traffic light system,” in *International Conference on Advances in ICT for Emerging Regions (ICTer)*. IEEE, 2014, pp. 91–95.

[11] K. Yuan, V. L. Knoop, and S. P. Hoogendoorn, “Optimal dynamic green time for distributed signal control,” in *IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2016, pp. 447–451.

[12] E. Ricalde and W. Banzhaf, “A genetic programming approach for the traffic signal control problem with epigenetic modifications,” in *European Conference on Genetic Programming*. Springer, 2016, pp. 133–148.

[13] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, “Recent development and applications of SUMO - Simulation of Urban MObility,” *International Journal On Advances in Systems and Measurements*, vol. 5, no. 3&4, pp. 128–138, December 2012.

[14] A. Wegener, M. Piórkowski, M. Raya, H. Hellbrück, S. Fischer, and J.-P. Hubaux, “Traci: an interface for coupling road traffic and network simulators,” in *Proceedings of the 11th communications and networking simulation symposium*. ACM, 2008, pp. 155–163.

[15] L. Codeca, R. Frank, and T. Engel, “Luxembourg sumo traffic (lust) scenario: 24 hours of mobility for vehicular networking research,” in *Vehicular Networking Conference (VNC), 2015 IEEE*. IEEE, 2015, pp. 1–8.

[16] L. Bieker, D. Krajzewicz, A. Morra, C. Michelacci, and F. Cartolano, “Traffic simulation for all: a real world traffic scenario from the city of bologna,” in *Modeling Mobility with Open Data*. Springer, 2015, pp. 47–60.

[17] V. Mauro and C. Di Taranto, “Utopia,” *Control, computers, communications in transportation*, 1990.

[18] DLR, “Sumo wiki,” 2017, [accessed 18-July-2017]. [Online]. Available: http://sumo.dlr.de/wiki

[19] J. R. Koza, “Genetic programming: A paradigm for genetically breeding populations of computer programs to solve problems,” Stanford University, Computer Science Departament, California, USA, Tech. Rep. STAN-CS-90-1314, 1990.

[20] D. J. Montana, “Strongly typed genetic programming,” *Evolutionary computation*, vol. 3, no. 2, pp. 199–230, 1995.

[21] J. A. Sousa and E. Costa, “Epial-an epigenetic approach for an artificial life model.” in *ICAART (1)*, 2010, pp. 90–97.

[22] O. Chikumbo, E. Goodman, and K. Deb, “Triple bottomline many-objective-based decision making for a land use management problem,” *Journal of Multi-Criteria Decision Analysis*, vol. 22, no. 3-4, pp. 133–159, 2015.

[23] W. La Cava, K. Danai, L. Spector, P. Fleming, A. Wright, and M. Lackner, “Automatic identification of wind turbine models using evolutionary multiobjective optimization,” *Renewable Energy*, vol. 87, pp. 892–902, 2016.