Smart Traffic Lights: A First Parallel Computing Approach

D. Souravlias*, G. Luque†, E. Alba†, K.E. Parsopoulos*
*Department of Computer Science and Engineering,
University of Ioannina, GR-45110 Ioannina, Greece
Email: {dsouravl,kostasp}@cse.uoi.gr
†Universidad de Málaga, Andalucía Tech
E.T.S.I. Informática, Campus Teatinos,
29071 Málaga (España)
Email: {gabriel,eat}@lcc.uma.es

***Abstract*—Optimal traffic light scheduling is a fundamental problem in modern urban areas. It has severe impact on traffic flow management, energy consumption and vehicular emissions, as well as on urban noise. The vast number of traffic lights in modern cities increases the complexity of the scheduling problem and, at the same time, urgently needs for efficient algorithms that optimize the light cycle programs. In this work, we propose a solution for the traffic light scheduling problem by using Differential Evolution, and investigate the benefits of parallelism on this complex problem. For understanding the impact in the city, the popular micro-simulator SUMO is used. We evaluate our approach on close-to-reality problem scenarios consisting of two large urban areas located in the cities of Málaga, Spain, and Paris, France. Our results are promising and encourage further investigation of parallel approaches to enable scalability.**

I. Introduction

Urban traffic planning is a fertile area of Smart Cities to improve efficiency, environmental care, and safety, since the traffic jams and congestion are one of the biggest sources of pollution and noise. Traffic lights play an important role in solving these problems as they control the flow of the vehicular network in the city. These devices are positioned at road intersections, pedestrian crossings, and other locations to control conflicting flows of traffic and avoid possible accidents. At each intersection, all traffic lights are synchronized to carry out a sequence of valid phases periodically. Each phase consists of a combination of color's states and has a time span that vehicles are allowed to use a roadway. The assignment of the time span for each phase in the phase sequence of all intersections at an urban area is called a *traffic light plan*.

Finding an optimal traffic light plan is crucial for reducing the number of stops for red lights, thereby minimizing the travel time of vehicles through the road network. Intuitive examples are the well-known *green waves*, which facilitate a continuous traffic flow in one main direction. Reducing the travel time prevents drivers from time-loss and late arrivals, with subsequent economic impact. Also, it helps reducing the fuel consumption and CO_2 emissions while the vehicle is stopped at red lights.

The many obvious benefits of optimal traffic light scheduling have motivated a growing field of research related to automatic traffic control signals. A number of industrial solutions have been proposed for this problem, such as the Cross Zlín [1] and ATC [2]. These solutions focus on the real-time configuration of a single traffic light junction. Also, they require the existence of infrastructures that provide online information about the changing traffic situations. We here go in a different direction, because the increasing number of vehicles requires the transition from the local control of a single intersection to a holistic approach considering a large urban area, and because optimizing the existing traditional traffic lights rests still much unexplored.

This holistic approach is only possible by using advanced computational resources and techniques due to the complexity of the problem, which is twofold. Firstly, the problem usually offers huge theoretical search spaces. For example, a simple intersection with 8 traffic light phases represents 55^8 (more than 8.3×10^{14}) possible solutions. Secondly, there is no closed mathematical formulation of the problem to assess the quality of candidate traffic lights configurations. Thus, the utilization of simulators is necessary. However, simulators are usually time-consuming, typically requiring from seconds up to a few minutes per simulation. Hence, new and efficient algorithmic tools become indispensable in real-world scenarios.

The present paper contributes in various aspects of the traffic light scheduling problem. Firstly, we propose an approach based on an established algorithm, namely Differential Evolution [3]. Several variants of the algorithm are tested to distinguish the most competitive ones. Besides, parallel versions are studied in order to improve efficiency and solve large, realistic instances. Finally, the proposed approach is evaluated on two large, real-world urban scenarios for the cities of Málaga (Spain) and Paris (France). The latter instance involves the optimization of more than 375 traffic lights, while the largest instances tackled in previous works [4], [5] were restricted to cases of up to 190 traffic lights.

The rest of the paper is structured as follows: in Section II we describe the related mathematical model and review related works. Then, we present the considered approaches in Section III. Section IV discusses the experimental results on the test cases of Málaga and Paris. Finally, Section V concludes

the paper and provides guidelines for future work.

II. Traffic Light Scheduling

Metaheuristics have been widely used to tackle traffic light scheduling problems. Early attempts were mostly based on Genetic Algorithms (GAs). The first study appeared in [6], where a GA was employed to optimize the timing of the traffic light cycles of nine intersections located in the city of Chicago (IL), USA. The authors proposed further investigation of GAs on larger problem instances. In [7], the authors studied the reactions of drivers to changes of the traffic lights timing. Their approach used a GA and it was evaluated on a case study of the city of Chester, UK. A GA was used also in [8] to optimize traffic light cycle programs. In this work, the authors assumed that the traffic lights timing of each intersection works independently of other intersections. They tested their approach on a test case of a commercial area of the city of Santa Cruz, Spain. Another work involving the application of GAs on a traffic light scheduling problem appeared in [9]. The proposed approach tackled the problem of controlling the traffic lights timing for vehicles and pedestrians under a dynamic traffic load situation.

Recently, there has been a number of works focusing on the application of the Particle Swarm Optimization (PSO) algorithm on finding optimal traffic light schedules. In [10], PSO was employed to train a fuzzy logic controller installed at each intersection. Specifically, PSO was used to train the membership functions and the rules of the controller, targeting to detect the optimal duration of the green signal for each phase of the traffic lights. In [11], the authors proposed a PSO algorithm to discover isolation niches on a traffic light scheduling problem. The proposed approach was evaluated on a small problem instance, consisting of an one-way road with two intersections. This work focused on the potential of the algorithm to maintain its diversity, without trying to gain deep insight on the problem at hand.

A multi-objective PSO algorithm that employed a predictive model control strategy to optimize traffic light cycle schedules was studied in [12]. The proposed algorithm was evaluated on an urban network consisting of 16 intersections and 51 links. In [4], [5], PSO was proposed for computing the optimal traffic light cycle programs. The main objectives of these works were the maximization of the number of vehicles that reach their destinations, as well as the minimization of the total trip time of the vehicles. The evaluation of the cycle programs was based on the popular microscopic SUMO simulator. The proposed algorithm was assessed on small/medium urban areas located in the cities of Málaga and Sevilla, Spain, and in Bahía Blanca, Argentina.

More recently, PSO algorithms were used for detecting traffic light cycle programs, aiming at the reduction of fuel consumption and vehicular emissions in metropolitan areas [13], [14]. These approaches followed a traffic emission model standardized by the European Union reference framework. The proposed algorithm achieved significant improvements in the considered objectives compared to traffic light cycle programs designed by experts.

A. Mathematical Formulation

We use here the mathematical model presented in [4], [5] for the traffic light scheduling problem. The considered problem has multiple objectives. The first objective is to maximize the number, V_{R}, of vehicles that reach their destination or, equivalently, minimize the number V_{NR} of vehicles that do not arrive at their destination, during a given simulation time T_{sim}. A second objective is to minimize the total trip time, T_{trip}, of the vehicles, which is equal to the sum of the trip times of all vehicles. The trip time refers to the simulation time individually consumed by each vehicle to arrive at its destination. Evidently, vehicles that fail to reach their destination consume the whole simulation time.

A third objective is to minimize the sum of stop and wait times of all vehicles, denoted by T_{sw}. The stop and wait time refers to the overall time that each vehicle individually has to stop at those intersections that have traffic lights in red color, thereby delaying its trip. A final objective is to maximize the ratio P of green and red colors in each phase state of all intersections, which is defined as follows:

$$P = \sum_{i=0}^{intr} \sum_{j=0}^{ph} d_{i,j} \, \frac{g_{i,j}}{r_{i,j}}, \tag{1}$$

where $intr$ denotes the number of all intersections; ph denotes the number of all phases; and $g_{i,j}$, $r_{i,j}$, denote the number of green and red signal colors, respectively, at intersection i and phase state j, with duration $d_{i,j}$. The minimum value of $r_{i,j}$ is set to 1 in order to prevent division by zero.

The intuition behind Eq. (1) lies in the effort to promote green traffic signals at intersections overburdened by traffic flow, and red traffic signals at intersections where low traffic flow is observed. Traffic lights with extended times in red color may overwhelm not only the intersection where they are located, but also neighboring intersections, creating extensive traffic flow problems in the city.

We combine all objectives into a single objective function formulated as follows:

$$f_{\mathrm{obj}} = \frac{T_{\mathrm{trip}} + T_{\mathrm{sw}} + V_{\mathrm{NR}} \, T_{\mathrm{sim}}}{V_{\mathrm{R}}^2 + P}. \tag{2}$$

It shall be noted that the quantities under minimization are placed in the numerator of Eq. (2), whereas the ones under maximization are placed in the denominator. Therefore, the overall problem is a global minimization task. The term V_{R} is squared to prioritize over all other terms as it represents the main (first) objective. Also, the number of non-arriving vehicles V_{NR} is multiplied by the simulation time T_{sim} to induce a penalization for this undersided scenario.

III. Proposed Approach

In this section, we briefly describe the employed SUMO simulator, the considered Differential Evolution and Particle

Swarm Optimization algorithms, as well as the proposed parallel model.

A. SUMO: Simulator of Urban Mobility

As already mentioned, simulation plays a crucial role in the assessment of optimization algorithms for the traffic light scheduling problem. *Simulator of Urban MObility* (SUMO) [15] constitutes a well established tool for this purpose. SUMO is an open-source, highly-portable micro-simulator used in a multitude of works. It requires a number of input files in XML format that contain information about the road scenarios to be simulated.

Specifically, there is a network file `.net.xml` that stores information about the form of the map, namely nodes, edges, and connection links among them. The route file `.rou.xml` holds information about the journey of a vehicle from a starting point (starting vertex) to an ending point (destination vertex) as well as all intermediate points visited by the vehicle. The `.add.xml` files contain additional information about the map or the traffic lights. Finally, the `.tripinfo.xml` contains information used to evaluate traffic light cycle programs. For instance, it contains information about the vehicles' departure and arrival times that are used to compute each vehicle's total trip time.

A candidate solution vector (traffic light cycle program) is forwarded to SUMO, which in turn computes its objective value after a simulation procedure. SUMO starts the simulation after transforming the input vector and the information contained in its data files into real-world objects such as vehicles, intersections, traffic lights, etc. When the simulation is completed, SUMO returns all the information that is necessary for the computation of the objective function value of the specific traffic light cycle program for the city. Note that a single SUMO call is adequate for computing the corresponding function value because SUMO works in a deterministic way. Deterministic traffic simulators are preferable to stochastic ones as they acquire similar results at considerably lower computational cost [8].

B. Differential Evolution

Differential Evolution (DE) is a well-studied population-based algorithm used for continuous optimization. It was introduced by Storn and Price in [3] and, since then, it has gained increasing popularity [16]. The DE algorithm employs a population of N candidate solutions,

$$P = \{x_1, x_2, \ldots, x_N\},$$

where each n-dimensional vector x_i is called an *individual*. The algorithm begins with an *initialization phase*, where the individuals are randomly (usually uniformly) initialized over the corresponding search space $\mathbb{S}$. The cornerstone of the algorithm is the *exploration phase* where the individuals iteratively probe the search space by sampling new points through *mutation*, *crossover*, and *selection* operators.

At each iteration g, a mutated vector v_i is generated for each individual x_i by combining other individuals of the population. In this work, we consider the following well-known mutation operators:

$$\text{DE1}: v_i^{[g+1]} = x_{\text{best}}^{[g]} + F\left(x_{r_1}^{[g]} - x_{r_2}^{[g]}\right), \tag{3}$$

$$\text{DE2}: v_i^{[g+1]} = x_{r_1}^{[g]} + F\left(x_{r_2}^{[g]} - x_{r_3}^{[g]}\right), \tag{4}$$

$$\text{DE3}: v_i^{[g+1]} = x_i^{[g]} + F\left(x_{\text{best}}^{[g]} - x_i^{[g]}\right) + F\left(x_{r_1}^{[g]} - x_{r_2}^{[g]}\right), \tag{5}$$

$$\text{DE4}: v_i^{[g+1]} = x_{\text{best}}^{[g]} + F\left(x_{r_1}^{[g]} - x_{r_2}^{[g]}\right) + F\left(x_{r_3}^{[g]} - x_{r_4}^{[g]}\right), \tag{6}$$

$$\text{DE5}: v_i^{[g+1]} = x_{r_1}^{[g]} + F\left(x_{r_2}^{[g]} - x_{r_3}^{[g]}\right) + F\left(x_{r_4}^{[g]} - x_{r_5}^{[g]}\right), \tag{7}$$

where $x_{\text{best}}^{[g]}$ denotes the individual with the best function value at generation g. The indices,

$$r_j \in \{1, 2, \ldots, N\} \setminus \{i\}, \qquad j = 1, 2, \ldots, 5,$$

are randomly selected and mutually different. The *differential weight* $F \in [0, 2]$ is a user-defined constant that controls the degree of expansion towards the directions defined by the difference vectors.

Mutation is followed by crossover, where a trial vector u_i is produced for each individual x_i in the following way:

$$u_{ij}^{[g+1]} = \begin{cases} v_{ij}^{[g+1]}, & \text{if } \; R(j) \leqslant CR \;\; \text{or} \;\; j = RI(i), \\ x_{ij}^{[g]}, & \text{otherwise,} \end{cases} \tag{8}$$

where $CR \in [0, 1]$ is a user-defined scalar called the *crossover probability*; $R(j) \in (0, 1)$ is a random number uniformly selected for each direction component, $j = 1, 2, \ldots, n$; and $RI(i) \in \{1, 2, \ldots, n\}$, is a random integer uniformly selected for each individual x_i of the population.

Finally, selection decides whether the trial vector replaces the corresponding original individual. Specifically, the replacement occurs if the trial vector improves in function value the original individual, i.e.,

$$x_i^{[g+1]} = \begin{cases} u_i^{[g+1]}, & \text{if } \; f\left(u_i^{[g+1]}\right) < f\left(x_i^{[g]}\right) \\ x_i^{[g]}, & \text{otherwise.} \end{cases} \tag{9}$$

The DE algorithm has been shown to be sensitive on its parameters N, F, and CR. Therefore, appropriate parameterization has significant impact on its performance and it is highly dependent on the considered problem. A concise presentation of research related to the DE algorithm can be found in [16].

C. Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a popular population-based algorithm used for numerical optimization, initially proposed by Eberhart and Kennedy [17]. The inspiration behind the algorithm originates from the collective behavior of socially organized living organisms. Up-to-date a significant amount of work has been devoted to the theoretical and empirical investigation of PSO [18], [19].

PSO employs a population, called a *swarm*, of N candidate solutions,

$$P = \{x_1, x_2, \ldots, x_N\},$$

where each vector x_i is called a *particle*. Initially, the n-dimensional particles $x_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ are randomly initialized within the search space $\mathbb{S}$. Then, each particle probes the search space iteratively, retaining in memory the best position it has ever discovered, denoted by $p_i = (p_{i1}, p_{i2}, \ldots, p_{in}) \in \mathbb{S}$. The movement of each particle is conducted by adding to its current position an adjustable position shift, called *velocity*, and denoted as $v_i = (v_{i1}, v_{i2}, \ldots, v_{in})$.

Thus, at iteration g, each particle updates its position according to:

$$x_{ij}^{[g+1]} = x_{ij}^{[g]} + v_{ij}^{[g+1]}, \tag{10}$$

where,

$$v_{ij}^{[g+1]} = \omega v_{ij}^{[g]} + c_1 R_1 \left(p_{ij}^{[g]} - x_{ij}^{[g]}\right) + c_2 R_2 \left(p_{\text{best},j}^{[g]} - x_{ij}^{[g]}\right), \tag{11}$$

where $p_{\text{best}}^{[g]}$ is the best particle of the swarm; ω is the inertia weight of the particle; c_1 and c_2 are the cognitive and the social parameters, respectively; and R_1 and R_2 are uniformly distributed random variables in the range $(0, 1)$.

At each iteration, each particle of the swarm updates also its best position as follows:

$$p_i^{[g+1]} = \begin{cases} x_i^{[g+1]}, & \text{if } f\left(x_i^{[g+1]}\right) < f\left(p_i^{[g]}\right) \\ p_i^{[g]}, & \text{otherwise} \end{cases} \tag{12}$$

In the present work, the inertia weight changes linearly throughout the optimization process according to the following rule:

$$\omega = \omega_{\max} - \frac{(\omega_{\max} - \omega_{\min})\, g}{g_{\max}}, \tag{13}$$

where $\omega_{\min}$ and ω_{max} define its range, g is the iteration counter, and $g_{\max}$ is the maximum number of iterations. At the beginning of the optimization process, Eq. (13) allows the inertia weight to take high values, thereby promoting exploration, whereas as ω reduces, better exploitation properties are achieved.

As suggested in [4], [5], the update of the velocity can be properly modified to tackle combinatorial problems. Specifically, each element of the velocity vector is transformed as follows:

$$v_{ij}^{[g+1]} = \begin{cases} \lfloor v_{ij}^{[g+1]} \rfloor, & \text{if } R \leqslant \lambda, \\ \lceil v_{ij}^{[g+1]} \rceil, & \text{otherwise,} \end{cases} \tag{14}$$

where $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ are the floor and ceiling functions, respectively, and R is a random number uniformly distributed in the range $(0, 1)$. The parameter λ determines the probability of using the floor or ceiling function in the computation of the velocity. In our study, its value is set to 0.5. A comprehensive presentation of the PSO algorithm can be found in [19].

D. Solution Enconding

Following the suggestions of previous works [4], [5], each direction component of the solution vector represents a phase duration of one state of the traffic lights of a particular intersection. Specifically, each state of each phase duration is encoded via an integer number, belonging to a simple vector of integers. The proposed encoding is desirable for various reasons. Firstly, the SUMO simulator itself employs integer numbers to represent the discrete time steps of the simulation procedure. Therefore, the mapping between the phase duration used by the data structures of SUMO and the solution vector is simplified. Secondly, the employed population-based algorithms can take into consideration the interdependence of variables, representing traffic lights of the same intersection as well as traffic lights of different intersections that exhibit high proximity.

Regarding the initialization of DE and PSO, the candidate solutions are initialized in the range $[5, 60]$. Each value in this range represents the time units (in seconds) that the corresponding traffic light will keep the same signal color, in the case of red and green colors. As for the amber signal color, we set its time interval to a constant value (4 seconds), which remains unchanged during the optimization. The proposed interval is selected according to various real-world traffic light scenarios provided by the City Council of Málaga (Spain).

The two algorithms are typically used in continuous optimization, where solutions consist of real numbers. In this work, the considered problem belongs to the class of combinatorial optimization problems. In order to tackle it, both DE and PSO use a proper rounding of the solution vector at each iteration, thus converting it to a vector of integer values. This conversion is necessary also for implementation purposes, since the SUMO simulator takes as input only integer values.

E. Parallel Model

Hard optimization problems require high computational times. In the case of metaheuristics, the most costly part of their execution is the evaluation of the objective function. Parallel metaheuristics have been shown to mitigate this deficiency [20]. Two parallelization strategies for population-based metaheuristics have prevailed in the recent years. The first one involves the *parallelization of operations*, where the operations applied to each individual of the population are performed in parallel. The second one, called *parallelization of population*

TABLE I
MÁLAGA AND PARIS PROBLEM INSTANCES.

Problem instance	Number of traffic logics	Number of traffic lights	Number of vehicles
Málaga	56	190	1200
Paris	70	378	1200

TABLE II
PARAMETERS OF THE ALGORITHMS.

Alg.	Param.	Description	Value(s)
DE	N	Population size	50
	F	Differential weight	{0.5, 0.7}
	CR	Crossover probability	{0.05, 0.1}
PSO	N	Population size	100
	c_1	Cognitive parameter	2.05
	c_2	Social parameter	2.05
	$\omega_{\min}$	Min. inertia weight	0.1
	$\omega_{\max}$	Max. inertia weight	0.5

approach, proposes the division of the population into parts, each one executed on a different processing unit.

In this work, we adopt the parallelization of population approach, where the function evaluations of the population of the algorithm are distributed evenly among the available processors. Also, we employ a simple master/slave parallelization model, where the main algorithm runs on the master node and the slave nodes play a pivotal role in the computation process. Specifically, the function evaluations of the population are divided into equal parts, each one assigned to a slave node running on a single processor. At each iteration, the algorithm requires the function evaluation of the individuals. First, the master sends the solution vectors (individuals) to the assigned slaves. Then, each slave computes the function values of the assigned individuals by performing a subsequent number of SUMO calls. Note that each function evaluation requires a single execution of SUMO. Eventually, each slave returns the computed function values back to the master, where they are further exploited for the next iteration of the algorithm.

The main reason for parallelizing the computation of the individuals is due to the high computation times required for each simulation run of SUMO. For example, we noticed that a single execution of SUMO on the Málaga and Paris problem instances required, on average, 1 s and 1.5 s, respectively, using a modern computation environment. Note that the employed parallel model does not affect the quality of the detected solutions (nor it changes the search model) compared to the serial algorithm. It is only a technological way to obtain the same quality of results, although in significantly less time.

IV. EXPERIMENTAL RESULTS

Our parallel approach is particularly suitable for tackling real-world urban scenarios. For this reason, we evaluated the proposed approach on two close-to-reality problem cases consisting of large metropolitan areas located in Málaga (Spain) and Paris (France). The considered problem cases were created by extracting information from real digital maps. The first problem instance involves the optimization of 190 traffic lights, whereas the second one contains 378 traffic lights. Previous works [4], [5] tackled problem instances with no more than 190 traffic lights. The dimension of each problem case is equal to the number of the traffic lights it comprises. More details on our instances can be found in Table I.

The experimental evaluation was conducted on the `saw` cluster of the Sharcnet consortium (`http://www.sharcnet.ca`). The parallel implementations were based on the OpenMPI project (`http://www.open-mpi.org`). Regarding the simulation procedure, each vehicle started its own trip from a starting point to a destination with a maximum speed of 50 km/h. This speed limit is typical in urban areas. The simulation time was set to 2200 seconds (iterations of microsimulation) for the Málaga instance and 3400 seconds for the Paris instance, as it consisted of a larger number of traffic lights. The simulation was conducted by executing the traffic simulator SUMO release 0.19.0. for Linux.

The first objective of the experiments was to compare the different variants of the DE algorithm on both problem instances. The population size of the algorithm was set to 50 individuals, each conducting 600 iterations, resulting in a computational budget of 30000 function evaluations overall. Each function value was obtained by a single simulation run of SUMO. We considered two distinct values for the differential weight parameter $F \in \{0.5, 0.7\}$, as well as for the crossover probability $CR \in \{0.05, 0.1\}$. Details for the parameterization of the DE algorithm are given in Table II.

For each problem case and DE variant, 10 independent experiments were conducted and the best detected solution was recorded. Each independent experiment required 10 to 13 hours of simulation, and it was terminated as soon as the maximum number of function evaluations was attained. For each problem instance and algorithmic variant, the average of the obtained solution values and the standard deviation are reported in Tables III and IV. The best approach per problem case, i.e., the one with the lowest average function value is boldfaced.

To further facilitate the comparisons among the different DE variants, we performed pairwise Wilcoxon rank-sum tests between all pairs of the considered approaches. For each variant, we counted the number of *wins*, i.e., the number of comparisons that it outperformed another variant (achieved lower mean function value) with significance level 95%.

The approaches that appear boldfaced in Tables III and IV exhibited the highest number of wins. Specifically, the best approach on the Málaga instance, namely the one that used the DE3 operator and parameter values $F = 0.5$, $CR = 0.1$, won 18 different variants. The best approach on the Paris instance, namely the one that employed the DE1 operator and parameter values $F = 0.7$, $CR = 0.1$, won 17 different variants. Notice that the total number of comparisons among the different DE variants is equal to 19.

The number of rank-sum wins per DE variant is graphically

TABLE III
RESULTS OF DE ALGORITHM ON THE MÁLAGA INSTANCE.

OP	F	CR	Mean	StD
DE1	0.5	0.05	0.4908	0.0098
	0.5	0.1	0.4895	0.0085
	0.7	0.05	0.5010	0.0073
	0.7	0.1	0.4992	0.0049
DE2	0.5	0.05	0.5015	0.0096
	0.5	0.1	0.5051	0.0079
	0.7	0.05	0.5141	0.0115
	0.7	0.1	0.5150	0.0070
DE3	0.5	0.05	0.5030	0.0098
	0.5	**0.1**	**0.4809**	**0.0080**
	0.7	0.05	0.4979	0.0062
	0.7	0.1	0.4979	0.0079
DE4	0.5	0.05	0.4988	0.0091
	0.5	0.1	0.4986	0.0114
	0.7	0.05	0.5058	0.0091
	0.7	0.1	0.5250	0.0134
DE5	0.5	0.05	0.5095	0.0138
	0.5	0.1	0.5178	0.0100
	0.7	0.05	0.5188	0.0101
	0.7	0.1	0.5408	0.0091

TABLE IV
RESULTS OF DE ALGORITHM ON THE PARIS INSTANCE.

OP	F	CR	Mean	StD
DE1	0.5	0.05	0.7420	0.0117
	0.5	0.1	0.7487	0.0096
	0.7	0.05	0.7502	0.0109
	0.7	**0.1**	**0.7332**	**0.0052**
DE2	0.5	0.05	0.7684	0.0117
	0.5	0.1	0.7681	0.0193
	0.7	0.05	0.7686	0.0097
	0.7	0.1	0.7603	0.0098
DE3	0.5	0.05	0.7764	0.0125
	0.5	0.1	0.7627	0.0600
	0.7	0.05	0.7602	0.0092
	0.7	0.1	0.7366	0.0111
DE4	0.5	0.05	0.7466	0.0111
	0.5	0.1	0.7344	0.0088
	0.7	0.05	0.7624	0.0063
	0.7	0.1	0.7521	0.0092
DE5	0.5	0.05	0.7732	0.0097
	0.5	0.1	0.7675	0.0105
	0.7	0.05	0.7759	0.0140
	0.7	0.1	0.7878	0.0085

illustrated in Fig. 1 for the Málaga case and Fig. 2 for the Paris case. In both cases, there is an evident superiority of the DE1 operator, achieving 45 wins for the Málaga case and 53 wins for the Paris case. Also, DE3 and DE4 operators achieved high numbers of wins compared to DE2 and DE5 ones, which exhibited the worst performance. This fact is attributed to the global information incorporated into the three best-performing operators. In general, experimental evidence suggests that it is beneficial to use exploitation-oriented operators in order to rapidly achieve a sub-optimal solution, under restricted computational budget. Operators DE1, DE3, and DE4 fulfill this necessity since they take advantage of the globally best solution, which rapidly leads the population to sub-optimal solutions.

In a second round of experiments, the best-performing DE variant for each problem competed against the global best model of the PSO algorithm. Details about the employed PSO were given in Section III-C. Regarding the parameterization of PSO, the swarm size was set to 100 and the cognitive and social constants were equal to 2.05. The PSO algorithm used an initial inertia weight $\omega_{\max} = 0.5$ that was linearly decreased to the final value $\omega_{\min} = 0.1$. Recall that the inertia weight adjusts the exploitation/exploration capabilities of PSO. The selected parameter values are also reported in Table II.

The obtained average values and standard deviations of PSO are reported in Table V. Evidently, the best-performing DE variant surmounts the specific PSO approach on each problem case. For the Málaga instance, the best DE approach achieved mean function value equal to 0.4809 in contrast to the value 0.5081 achieved by PSO. Similar performance is observed for the Paris instance where DE achieved an average

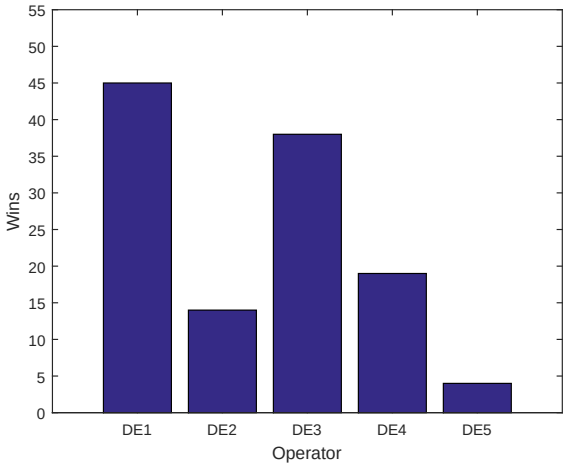

Fig. 1. Comparison of DE operators on Málaga instance

value 0.7332 whereas PSO achieved 0.7724. Although the differences between these values seem marginal, it shall be underlined that the improvement in the function value from one iteration to another during the optimization was usually observed in the second or third decimal digit using both DE and PSO algorithms. These small fitness differences represent an important difference in the solution space.

In the third round of the experiments, we evaluated the proposed parallel approach. The best-performing algorithm per problem was selected and implemented according to the parallelization of population approach. Specifically, we employed the DE variant that uses DE3 with $F = 0.5$, $CR = 0.1$ for

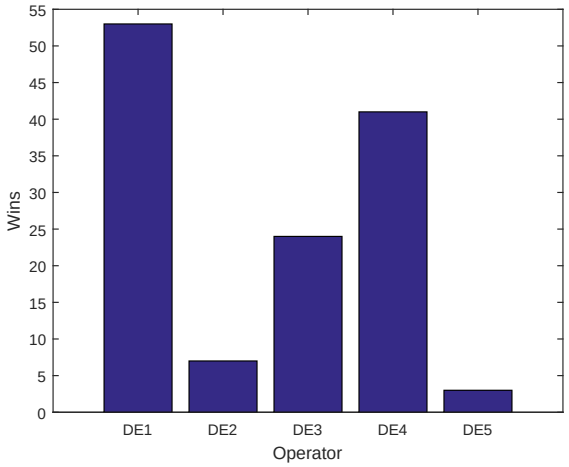

Fig. 2. Comparison of DE operators on Paris instance

TABLE V
RESULTS OF PSO ALGORITHM ON BOTH PROBLEM INSTANCES.

Problem instance	Mean	StD
Málaga	0.5081	0.0122
Paris	0.7724	0.0083

the Málaga problem and the one that uses DE1 with $F = 0.7$, $CR = 0.1$ for the Paris problem. The function evaluations were equally distributed among 2 and 3 nodes in an effort to speed up the computation process. Details about the proposed parallel model are given in Section III-E. For each problem and algorithmic variant, we recorded the execution time required to achieve a targeted objective function value, f_{trg}. For the Málaga problem, this value was set to 0.50, while for the Paris problem it was equal to 0.74. The specific values of f_{trg} were achievable at each independent experiment. As soon as the algorithm reached the targeted function value or a better one, it terminated its execution.

A significant gain metric for parallel implementations is the *speedup*, which computes the ratio between sequential and corresponding parallel execution times. Several speedup definitions are used in the literature. Here, we use the *weak speedup* definition [20]. The weak speedup s_m of a parallel algorithm using m processors is defined as follows:

$$s_m = \frac{T_{\text{seq}}}{T_m}, \tag{15}$$

where T_{seq} is the execution time of the sequential algorithm executed on a single processor, and T_m is the execution time of the parallel algorithm running on m processors.

Table VI reports the execution time (in hours) and the achieved speedup for the best serial algorithm per problem, parallelized across a number of nodes. We note that the reported number of nodes includes the master node, which runs the algorithm, and the slave nodes that conduct the function evaluations through parallel SUMO calls. Thus, 2 nodes refer to the serial execution of the algorithm (1 master and 1 slave) and it was used as a baseline for comparisons with the parallel approaches that assume higher number of nodes.

TABLE VI
RESULTS OF THE PARALLEL APPROACH.

Problem	Nodes	Time (hours)	Speedup
Málaga	2	1.90	–
	3	1.41	1.35
	4	1.10	1.73
Paris	2	9.28	–
	3	5.79	1.60
	4	5.29	1.75

For the Málaga instance, we observe that the sequential algorithm required 1.90 hours to achieve the considered f_{trg}. The approaches that exploited 3 and 4 nodes achieved lower execution times, namely 1.41 and 1.10 hours, respectively. In terms of speedup, Table VI shows a value of 1.35 for 3 nodes. The speedup value was further increased when 4 nodes were used. This result was expected since the function evaluations were distributed among a larger number of nodes, thus enhancing the performance of the algorithm. However, we notice that the percentage of increase in speedup values from 2 to 3 nodes is lower than the one from 3 to 4 nodes.

For the Paris instance, we observe that the parallel approach was even more beneficial. Specifically, the sequential algorithm required 9.28 hours, whereas the parallel approach using 3 and 4 nodes needed 5.79 and 5.29 hours, respectively. The corresponding speedup values were 1.60 for the 3-node case and 1.75 for the 4-node one. The longer execution time needed for the Paris instance is attributed to its problem size, which is almost twice the Málaga one, thereby leading to higher simulation times.

Thus, the proposed parallel model has evidently boosted the performance of the employed algorithms. This finding is highly desirable in real-world traffic light scenarios as the considered one. Nevertheless, we would like to underline the effect of the simulation procedure on the acquired results, since we noticed high deviation in the running times needed by the SUMO simulator.

V. CONCLUSIONS

Urban traffic planning has been recently established as an active research area of Smart Cities. It mainly copes with important problems such as traffic flow management, pedestrian safety, management of fuel consumption, reduction of vehicular emissions, and urban noise. Traffic light scheduling constitutes a crucial task in solving these problems.

We studied an approach based on the DE algorithm to optimize traffic light cycle programs. To the best of our knowledge, this is the first work that investigates the potential of the DE algorithm on the considered problem. We compared the performance of our approach in terms of solution quality with the PSO algorithm. Furthermore, we developed a parallel

version of our method in order to tackle more realistic instances, efficiently. The proposed methods have been assessed on two real-world scenarios, consisting of large metropolitan areas located in Málaga (Spain) and Paris (France).

Future research will focus on sophisticated parallel algorithmic schemes, such as Algorithm Portfolios, on demanding real-world problem instances using higher computational budgets. Also, experimentation with real-world scenarios that exhibit high levels of heterogeneity will be considered.

ACKNOWLEDGMENTS

This work is partially funded by project number 8.06/5.47.4142 in collaboration with the VSB-Technical University of Ostrava, the University of Málaga (Andalucía Tech), and the Spanish MINECO project TIN2014-57341-R (http://moveon.lcc.uma.es). The authors would like to thank the Shared Hierarchical Academic Research Computing Network (SHARCNET) consortium for offering the necessary computational resources.

REFERENCES

[1] CROSS Zlín, "eDaptiva: Full-featured Urban Traffic Management Center," Czech republic, Tech. Rep., 2015.

[2] Aldridge Traffic Controllers, "SCATS," Australia, Tech. Rep., 2015.

[3] R. Storn and K. Price, "Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[4] J. García-Nieto, A. Carolina Olivera, and E. Alba, "Optimal cycle program of traffic lights with particle swarm optimization," *IEEE Trans. on Evol. Comp.*, vol. 17, no. 6, pp. 823–839, 2013.

[5] J. García-Nieto, E. Alba, and A. Carolina Olivera, "Swarm intelligence for traffic light scheduling: Application to real urban areas," *Engineering Applications of Artificial Intelligence*, vol. 25, no. 2, pp. 274–283, 2012.

[6] N. M. Rouphail, B. B. Park, and J. Sacks, "Direct signal timing optimization: strategy development and results. technical report," in *XI Pan American conference in Traffic and Transportation Engineering*, 2000.

[7] F. Teklu, A. Sumalee, and D. Watling, "A genetic algorithm approach for optimizing traffic control signals considering routing," *Comput. Aided Civil and Infrastruct. Eng.*, vol. 22, no. 1, pp. 31–43, 2007.

[8] J. Sánchez, M. Galán, and E. Rubio, "Applying a traffic lights evolutionary optimization technique to a real case: las ramblas area in santa cruz de tenerife," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1, pp. 25–40, 2008.

[9] A. M. Turky, M. S. Ahmad, M. Z. M. Yusoff, and B. T. Hammad, "Using genetic algorithm for traffic light control system with a pedestrian crossing," in *Proceedings Fourth International Conference on Rough Sets and Knowledge Technology*, 2009, pp. 512–519.

[10] J. Chen and L. Xu, "Road-junction traffic signal timing optimization by an adaptive particle swarm algorithm," in *Proceedings 9th International Conference on Control, Automation, Robotics and Vision*, 2006, pp. 1–7.

[11] L. Peng, M. H. Wang, J. P. Du, and G. Luo, "Isolation niches particle swarm optimization applied to traffic lights controlling," in *Proceedings of the 48th IEEE Decision and Control Chinese Conference CDC/CCC*, 2009, pp. 3318–3322.

[12] S. Kachroudi and N. Bhouri, "A multimodal traffic responsive strategy using particle swarm optimization," in *Proceedings of the 12th IFAC Symposium on Transportation Systems*, 2009.

[13] J. García-Nieto, J. Ferrer, and E. Alba, "Optimising traffic lights with metaheuristics: Reduction of car emissions and consumption," in *IJCNN*, 2014, pp. 48–54.

[14] A. Carolina Olivera, J. García-Nieto, and E. Alba, "Reducing vehicle emissions and fuel consumption in the city by using particle swarm optimization," *Applied Intelligence*, vol. 42, no. 3, pp. 389–405, 2015.

[15] D. Krajzewicz, M. Bonert, and P. Wagner, "The open-source traffic simulation package of sumo," in *RoboCup 2006 Infrastructure Simulation Competition*, 2006.

[16] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, 2011.

[17] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings Sixth Symposium on Micro Machine and Human Science*, Piscataway, NJ, 1995, pp. 39–43.

[18] R. Poli, "Analysis of the publications on the applications of particle swarm optimisation," *J. Artif. Evol. App.*, no. 3, pp. 1–10, 2008.

[19] K. E. Parsopoulos and M. N. Vrahatis, *Particle Swarm Optimization and Intelligence: Advances and Applications*. Information Science Publishing (IGI Global), 2010.

[20] E. Alba, *Parallel Metaheuristics: A New Class of Algorithms*. Wiley-Interscience, 2005.