Comparative of Traffic Optimization with SUMO Using Genetic Algorithm and Particle Swarm Optimization

Martín Montes Rivera, Jorge Alonso Ramírez-Márquez,
Jesús Rafael Tavarez-Delgado

Universidad Politécnica de Aguascalientes,
Mexico

{martin.montes, jorge.ramirez, rafael.tavarez}@upa.edu.mx

Abstract. One of the main issues in urban transportation is traffic congestion, which wastes time and money resources. Aguascalientes city in Mexico is not an exception. Thus a research group integrated by Investel, Centro de Investigación en Matemáticas (CIMAT), Universidad Politécnica de Aguascalientes (UPA), among others has started build a solution for this type of problems. However, constructing new structures requiring a high investment is not always possible due to economic, demographic, and social reasons. As an alternative, traffic control techniques speed up vehicular traffic by controlling the time of traffic lights. Artificial intelligence is one approach to determining the optimal time for traffic light phases. In this work, we built a scenario with six traffic lights to test a Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) to optimize traffic control as an initial step for the responsive control of Aguascalientes traffic lights. The optimization approach uses fitness and cost functions that measure the stopping time for each car in the traffic flow. Both optimization algorithms identify the necessary periods by evaluating candidate solutions with simulations performed by the SUMO platform, which considers traffic jams, car collisions, and time delays. Finally, we compare the GA and PSO (PSO), identifying the best algorithm in this scenario. The proposed method in this work will allow us to test several algorithms optimizing traffic lights phases for specific scenarios, which eventually help us with the appropriate sensors to generate responsive traffic lights in different road junctions in Aguascalientes, Mexico.

Keywords: Traffic lights optimization, simulation of urban mobility, GA, PSO, road traffic control.

1 Introduction

Transportation has been a critical issue within human civilization. One transportation problem is traffic congestion. Traffic congestion appears when too many vehicles use standard transport in infrastructure with a limited capacity [1]. Traffic congestion is a problem in several countries, including Mexico. However, a solution may be to build new roads; this is not always possible due to social, environmental, and economic

characteristics [2]. Aguascalientes, founded in 1575, is a state with an area of 5600 km2, located in the north-center of Mexico [3]. The urban mobility of Aguascalientes has grown since then, adding new parks, bikeways, vehicular bridges, and overpasses to deal with the increase of vehicles and pedestrian traffic flow [4]. Constructions presented in [5–8] support the construction of bridges and overpasses of the city, modifying the traffic to reduce the transportation time and maintaining the existing infrastructure for vehicles.

As an alternative to infrastructure, the systems for vehicular traffic control use diverse computer methods, and transport management methods use modern management systems with telecommunication technologies [9].

The primary control measure for urban land transport is traffic lights at intersections. The fundamental objective of traffic lights was initially to guarantee the safe crossing of vehicles and people [1]. However, due to the progressive increase in users, it has become necessary to establish the appropriate methodologies to make their use more efficient.

A simulation is the imitation of processes or systems in the real world. Also, a simulation is an indispensable methodology for testing the solutions and analyzing a system's behavior [10].

The traffic models for their simulation imitate traffic densities and average speeds [2]. Thus, these models allow generating focused strategies to improve vehicular traffic. In addition, computer simulations can point out the relation between the logical values of the problem and the decision-making system [11].

This article proposes a method for comparing algorithms optimizing traffic lights to reduce traffic congestion. We use two numerical optimization algorithms, Genetic Algorithms (GA) and Particle Swarm Optimization (PSO), the most representative of evolutionary and swarm intelligence algorithms, respectively. All simulations in this work use the Simulation of Urban Mobility (SUMO) platform.

SUMO is a traffic simulation software that allows modeling road vehicles, public transport, and pedestrians. Among the applications carried out with SUMO are evaluating traffic lights' performance and algorithms for their control [12].

Several investigations about the SUMO traffic simulations involve models for traffic demand and optimization in places such as Luxembourg, Medellin, Surabaya, Bologna, and Bandung Casablanca, Milan, and Samara [13–22]. Some of the characteristics of this software are that it is open-source, allows the simulation of cars, and contains tools to import road networks and generate routes from different sources to reproduce realistic mobility patterns.

The approach proposed in this work allows repeating the method for different crossroads with traffic lights in Aguascalientes, Mexico, towards developing a responsive vial control.

Input parameters: N_P, I_T, $f(X_i)$, L_l, L_u, LS_l, LS_u, c_1, c_2, w
X_i=$\psi_i \in [L_l, L_u]$;
$V_i = 0$;
$C_i = f(X_i)$;
$G_P = C_i$;
$G_B = \min(G_B)$;
$P_i = X_i$;
$P_G = X_{argmin(f(X_i))}$;
while $k < k_{max}$ **or** $G_B \leq C_d$ **do**
  $V_i = w \cdot \mathrm{V_i} + c_1 \cdot R_1 \cdot (P_G - X_i) + c_2 \cdot R_2 \cdot (P_i - X_i)$; $V_i = V_i \in [LS_l, LS_u]$;
  $X_i = \mathrm{X_i} + V_i$; $X_i = \psi_i \in [L_l, L_u]$;
  $C_i = f(X_i)$;
  for i **from 1 to** I_T **do**
    if $G_P < C_i$ **then**
      $G_P = C_i$;
      $P_i = X_i$;
    end
  end
  $G_B = \min(G_P)$;
  $P_G = argmin\big(f(P_i)\big)$;
end

Algorithm 1. PSO algorithm.

2 Theoretical Framework

2.1 Particle Swarm Optimization

The Particle Swarm Optimization algorithm (PSO) is a population-based optimization method developed in 1995 by James Kennedy and Russell Eberhart [23]. PSO mimics the nature and social behavior of organisms in groups, such as the motion of birds when they group in flocks, the motion of fish in schooling, and the ant colonies [24]. Nowadays, PSO has become a popular numerical optimization algorithm, and several variations using regrouping, neighborhoods, and unique parameters have been proposed [24–26]. However, in this work, we use the most common variant in proposed 1998 with inertial momentum [27].

The steps of PSO are in Algorithm 1. Its parameters include the minimum and maximum boundaries of the position of particles $X_i = \psi_i \in [L_l, L_u]$ ($[X_{min}, X_{max}]$), their velocities ($V_i = V_i \in [LS_l, LS_u]$) (initialized in zero); the inertial parameter (w), the maximum iterations (k_{max}), coefficients (c_1 and c_2) for personal and social components, the cost function $f(X_i)$, the cost value (C_i) of X_i, the best-known positions for each particle ($P_i = X_i$) , the global best cost G_B and the global best position ($P_G = X_{argmin(f(X_i))}$). The procedure iterates until I_T reaches k_{max} or finding the desired cost value (C_d) [27].

```
Input parameters: S_P, P, M_P, T_S, N_G, P_M
Create random population P
Evaluation of fitness (F)
while ending condition is not met, do
   Select individuals with the best F
   Crossover the selected individuals
   Mutate offspring from the crossover
   Evaluate F of new individuals
   Insert new individuals into the population
   Erase individuals with the worst F
End
Return individual with highest F
```

Algorithm 2. GA algorithm.

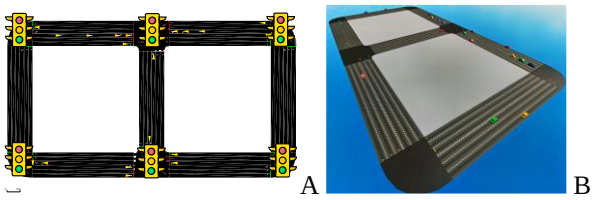


Fig. 1. Proposed Scenario. A: Schematic and B: Three-dimensional representations.

2.2 Genetic Algorithm

The genetic algorithm (GA) was developed between the 1960s and 1970s by John Holland, inspired by the natural selection principles proposed by Charles Darwin. The sections involved in the algorithm are population initialization, fitness calculation, crossover, mutation, and fitness selection [28–31].

Algorithm 2 describes GA. The GA parameters are population size (S_P), the tournament size (T_S), the number of generations (N_G), the mutation probability (P_M), the fitness function (F). The GA iterates until reaching the stopping conditions, such as the maximum generations N_G or the desired fitness value [32]. We use tournament selection, single-point crossover, and random mutation for this work.

3 Methodology

We start defining the optimization scenario using Python 3.8.10 with the last version of the SUMO library 1.1 and the plugging sumo-web3d 1.0. This scenario considers four possible situations in everyday crossroads with traffic lights: Go ahead, turn left, turn right, and U-turn. The map includes six crossroads with traffic lights, as shown in Fig. 1 for schematic and three-dimensional representations.

The traffic lights phases have two possible configurations. The first with states: 'GGGgGGGr,' 'yyyyyyyr,' 'GGGrGGGg,' and 'yyyryyyy.' The second with states: 'GGGgggrrrrGGGGg', 'yyyyyyrrrryyyyy', 'rrrrrrGGGgGrrrr', 'rrrrrryyyyyrrrr'.

The scenario proposed in Fig. 1 allows simulating complex behaviors in the SUMO library like:

- Covariance of traffic lights: Times for each traffic light depend on others.
- Traffic jams: Accumulation of cars when setting wrong traffic lights.
- Cars collisions: Accidents of cars when setting wrong times in the yellow light.
- Delay times: Cars must wait when the traffic light or a jam stops them.
- The vehicles' randomness: The decisions of cars at a crossroad are random.

3.1 Scenario Optimization with GA and PSO

Once the scenario was defined, we connected it with the optimization algorithms, following the definitions in sections 2.1 and 0. This implementation uses Python with the Traci library, which SUMO uses to manage connections to its simulator and access the resulting data.

The schema we used for optimizing the traffic scenario depends on numerical optimization, like the one performed with GAs and PSO. The parameters that the optimization algorithms tune were initially 24, which are the times for each phase of the six traffic lights on the map. However, we found that the phases that include yellow lights are constants in the optimization since they represent the required time for each driver to realize that the current light is about to change. Therefore, after eliminating each phase associated with the yellow light, we reduce the optimization to 12 dimensions, two phases per traffic light.

After identifying the parameters to optimize, we focus on the goal of this work, which is to optimize the times on traffic lights to reduce the stopped time for each car in the traffic flow.

Cost Function.

For preparing our optimization function, we consider reducing the vehicles' stopping time. Based on that idea, the accumulated stopped time for the vehicles indexed with i depends on the 12 times used in the phases on the traffic lights in the optimization, as shown in equation (1):

$$t_{s,i} = f(t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}, t_{12}). \quad (1)$$

Thus, our proposed cost function in equation (2) is the summation of all the vehicles' accumulated stopping times in the simulation. We will consider a minimizing problem in the optimization since the work goal is to reduce the stopping time:

$$Cost(t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8, t_9, t_{10}, t_{12}) = \sum_{i=1}^{n} t_{s,i}. \quad (2)$$

GA implementation.

Once the GA follows the definitions in section 0, we must configure its input parameters which include:

- Number of generations (N_G): We changed the parameter N_G to the number of evaluations (N_E). Because the GA and the PSO must evaluate the same number of

times the cost function to obtain a comparative result, the N_E, includes the evaluations used in the GA to generate the initial population and the evaluations of each generation. In this work, we set N_E=300, for 250 seconds per run in the GA.

- Population size (S_P). The population size affects the GA results since it represents the initial fitness evaluations to explore the search space [33]. Thus, we identify the best population size from 5-205 chromosomes with steps of 10 chromosomes or 1.66667% up to 68.33333% of the allowed cost function evaluations. Determining S_P with reliability implies different scenarios for validation and a technique for tunning like K-fold cross-validation. However, this is not the goal of the work, and we are interested only in this scenario. Therefore, when the scenario changes, we must repeat the tunning process.
- Tournament size (T_S). Tournament size affects the convergence speed in GAs, and a wrong value could result in premature convergence [33, 34]. Thus, we selected values from 3-100 chromosomes or crossover probability from 1% to 30% with steps of 5 chromosomes. We follow the same process described for S_P for determining T_S. Therefore, when the scenario changes, we must repeat the tunning process.
- Mutation Probability (P_M). The mutation mechanism helps introduce new information in GAs and avoid premature convergence. However, if P_M is high, an excess of noise affects the offspring. Thus, the literature recommends at much to 5% of probability [32]. Therefore we use 5% in this research.

The best GA parameters for our scenario test a total of 600 configurations for tournament size and the number of chromosomes.

PSO implementation.

Once the PSO follows the definitions in section 2.1. We must configure its input parameters which include:

- Number of iterations (I_T): We changed the parameter I_T to the number of evaluations (N_E). Because as mentioned before, PSO and GA must evaluate the same number of times the cost function to obtain a final comparative result. Again, we set the N_E=300, for 250 seconds per run. If the scenario changes, we must repeat the tunning process as mentioned in the GA implementation.
- Maximum values for the position [X_{min}, X_{max}]: We set them to [0.3,1.0], allowing reach the original from 30% to 100% of the original value. The minimum value selected allows avoiding cars collisions.
- Number of particles (N_P): Affects the PSO in the number of cost function evaluations per iteration. Thus, we identify bests values using 5-205 with steps of 10 particles, representing 1.66667% up to 68.33333% of the allowed evaluations. If the scenario changes, we must repeat the tunning process as mentioned in the GA.
- Inertial parameter (w): Intertial behavior allows controlling exploration of the search space like N_P with fixed N_E. Thus, we selected the best value in our scenario using 0.9-1.2 with steps of 0.1 units as recommended in the work that proposes this behavior.

- Personal and social components (c_1, c_2): We set these components equal to 2.0 as recommended by the original algorithm.

4 Results

This section shows all the algorithms and simulations we test in this work. The computer we used to carry out all the experiments is a Microsoft Windows 10 Pro system with an Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz, 3401 Mhz with four central processors and RAM 16 GB, with motherboard ASUS Z170M-PLUS, and KINGSTON SUV400S37480G solid-state disk. All the results in this section consider the minimizing cost function proposed that measures the accumulated delay time for all the vehicles during the simulation, as mentioned in equation (2).

4.1 Results GA

We tested a total of 460 configurations with the GA in 30.39857 hours, with an average time of 237.90183 seconds per configuration, obtaining as the best result 33 seconds of accumulated delay time for 436 vehicles included in the SUMO simulation of 1000 seconds.

Fig. 2 shows the Surface Fitting, concerning the number of chromosomes, the tournament size, and the accumulated delay time for all the vehicles in the simulation. The blue values are the best ones with lower accumulated delay time. The best configuration is in the blue point at $(S_p = 85, Ts = 58, BestFitness = 33)$.

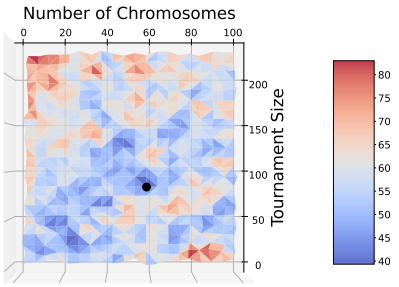


Fig. 2. Surface with all configurations for the GA and its fitness. Best configuration $(S_p = 85, Ts = 58, BestFitness = 33)$.

Fig. 3 shows the worst and best fitness across generations, in red and blue, respectively, for the GA's best configuration. The charts' generations are 110 since the algorithm stops after 300 fitness evaluations, as stated in section 3.1.

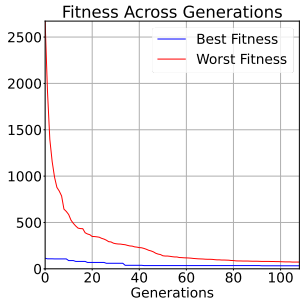


Fig. 3. Fitness across generations for the GA with the best configuration.

Table 1 shows the times of the best candidate solution of GA, including the two phases duration (phase of green light and phase of red light) on each of the six traffic lights in our scenario. This configuration produces 33 seconds of delay for 436 vehicles in 1000 seconds of simulation in SUMO.

Table 1. Time in phases for each traffic light obtained with the GA best candidate solution using the best configuration ($S_p = 85, Ts = 58, BestFitness = 33$).

Traffic Light	Phase with Green Light (seconds)	Phase with Red Light (seconds)
1	27.15134865	29.84865135
2	41.18746251	15.81253749
3	27.17673005	29.82326995
4	41.15427277	15.84572723
5	41.66752012	15.33247988
6	22.2938168	34.7061832

4.2 Results PSO

We tested a total of 69 configurations with the GA in 5.38600 hours, with an average time of 281.00336 seconds per configuration, obtaining as the best result 15 seconds of accumulated delay time for 436 vehicles included in the SUMO simulation of 1000 seconds.

Fig. 4 shows the Surface Fitting, concerning the number of particles, the inertial coefficient, and the accumulated delay time. Blue values are the best ones with lower accumulated delay time. The best configuration is in the blue point ($N_P = 55, w = 0.9, BestCost = 15$).

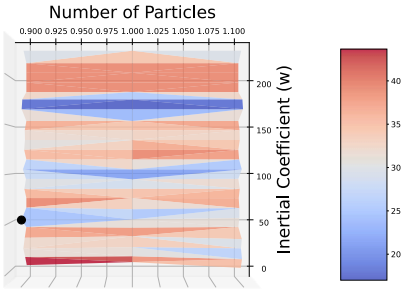


Fig. 4. Surface with all configurations for the PSO. Best configuration is ($N_P = 55, w = 0.9, BestCost = 15$).

Fig. 5 shows the worst and best cost across iterations, in red and blue, respectively, for the best configuration obtained in our scenario using the PSO. The charts' generations are five since the algorithm stops at this point after 300 fitness evaluations, as stated in section 3.1 for applying the same cost evaluations in GA and PSO algorithms.

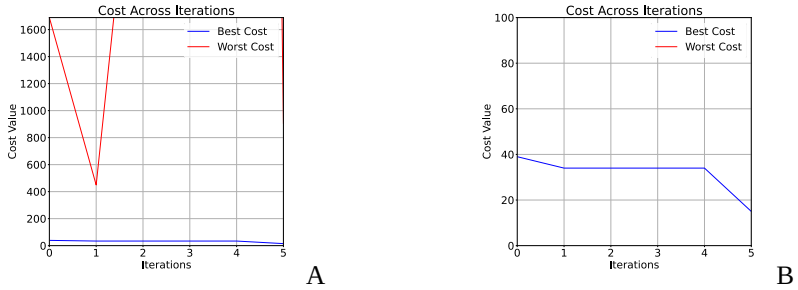


Fig. 5. Cost across iterations for the PSO with the best configuration. A: window between the worst initial cost and the best cost. B: window between the best initial cost and the best cost.

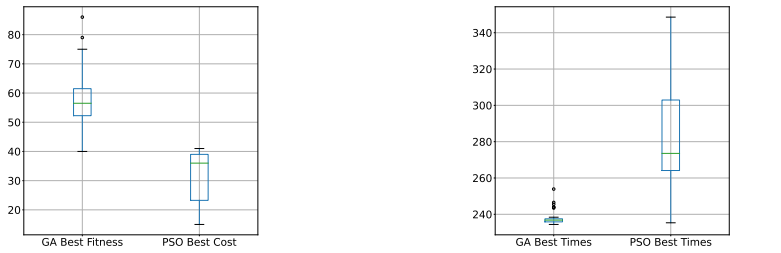
4.3 Comparison of GA and PSO Results

In this work, we are interested in determining the best algorithm to optimize times in traffic lights using the SUMO library for our proposed scenario with six traffic lights. We test the algorithms and evaluate them based on two critical variables, quality of best solution and execution time for obtaining it. Since we evaluated more GA than PSO configurations, the comparison results in this section consider only the best 30 configurations randomly sampled from GA and PSO configurations.

Fig. 6 Shows two box plots obtained from 30 random samples of the tried configurations. The first one (A) considers the fitness and cost values, and the second one (B) considers the execution times.

A box plot shows that the GA fitness has a lower variance than the cost value of the PSO when varying its parameters. However, the PSO obtained better quality solutions than GA in the majority of the configurations.

B box plot shows a lower variance in execution times per generation in the GA than in iterations of the PSO. However, this is related to how the algorithm works; the number of particles changes the number of simulations carried out by SUMO per iteration. On the other hand, changing the number of chromosomes or the mutation percentage does not modify the GA's number of simulations per iteration.



A B

Fig. 6. Comparison between GA and PSO algorithms with 30 random samples of the tried configurations. A: Comparison between fitness and cost values. B: Comparison between execution times.

After analyzing the box plot comparisons, we found that PSO produces better quality solutions than GA, but it requires more time per simulation for doing it. Even though PSO requires more time per iteration, we base our analysis on cost evaluations in this study. Thus, we reduce the number of iterations by increasing the number of particles, and even with that, the PSO produces better quality solutions. The ANOVA test in **Table 2** supports better quality solutions of PSO, with F-score = 102.95786 and P-value = 1.803639E-14, and lower execution times in GA, with F-score = 62.66525 and P-value = 8.46629E-11.

Table 2. Values of validation for the conclusion based on ANOVA.

ANOVA values	Fitness and Cost values.	Execution times in GA and PSO.
F-score	102.95786	62.66525
P-value	1.803639E-14	8.46629E-11

5 Conclusions

This work compares the GA with the PSO algorithm in quality and execution times when optimizing delay times of vehicles by changing phase times on traffic lights, which are crucial when optimizing traffic light controllers. All the algorithms and simulations we carry out use Python as a programming language with the open-source SUMO (Simulation of Urban Mobility) library.

We use the accumulated delay in cars in a SUMO simulation to optimize the times of traffic lights. We test the variation of input parameters on each algorithm and simulate their results, maintaining 300 as the maximum number of cost evaluations.

As a result, we found that PSO produces better quality solutions than the GA but requires more time per iteration when optimizing. However, we reduced the number

of iterations by increasing the number of particles, basing our analysis, but the PSO remains obtaining the best quality solutions.

We support our analysis on box plots with the ANOVA test, obtaining that the PSO has better quality solutions, with F-score = 102.95786 and P-value = 1.803639E-14. The GA has lower execution times per iteration, with F-score = 62.66525 and P-value = 8.46629E-11.

The main contribution of our work is a methodology for optimizing traffic lights based on accumulated delay times. First, we evaluate the results of the proposed method on a specific scenario.

5.1 Future Work

In this work, we probe sufficient evidence that the PSO produces better results in optimizing traffic lights for a specific scenario. However, both algorithms must test several more scenarios to support that one is better than the other. On the other hand, other algorithms could test our approach.

References

1. Papageorgiou, M.: Overview of Road Traffic Control Strategies. IFAC Proc. Vol. 37, 29–40 (2004). https://doi.org/10.1016/S1474-6670(17)30657-2.
2. Bellemans, T., De Schutter, B., De Moor, B., Bellemans, T., De Schutter, B., De Moor, B.: Models for traffic control. Delft Univ. Technol. Fac. Inf. Technol. Syst. J. A. 43, 13–22 (2002).
3. Gómez Serrano, J., Delgado Aguilar, F.J.: Aguascalientes: Historia breve. Fondo de Cultura Económica, Mexico City (2016).
4. H. Ayuntamiento de Aguascalientes 2019-2021: INICIA MUNICIPIO OBRAS DE REHABILITACIÓN DE LA AVENIDA MARIANO HIDALGO.
5. Dávalos, T.: Anuncian más obras públicas para Aguascalientes - El Sol del Centro | Noticias Locales, Policiacas, sobre México, Aguascalientes y el Mundo.
6. Granados, F.: ¿Qué sigue para la movilidad urbana en Aguascalientes?/ Agenda urbana - LJA Aguascalientes.
7. Newsweek: Inician obras de construcción de paso a desnivel de Av. Aguascalientes y Fracc. Parras | Newsweek México.
8. Flores Nieves, A.J.: Pendiente en Aguascalientes, el próximo puente en Barberena Vega - LJA Aguascalientes.
9. Pamula, T.: Road traffic parameters prediction in urban traffic management systems using neural networks. Transp. Probl. 6, 123–128 (2011).
10. Banks, J.: Introduction to simulation. WSC'99. 1999 Winter Simul. Conf. Proc. 1, 7–13 (1999). https://doi.org/10.1109/WSC.1999.823046.
11. Fedorko, G., Rosova, A., Molnár, V.: The application of computer simulation in solving traffic problems in the urban traffic management in Slovakia. Theor. Empir. Res. Urban Manag. 9, (2014).

12. Alvarez Lopez, P., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.-P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., Wießner, E.: Microscopic Traffic Simulation using SUMO. 2019 IEEE Intell. Transp. Syst. Conf. 2575–2582 (2018).
13. Krajzewicz, D.: Traffic Simulation with SUMO – Simulation of Urban Mobility. Int. Ser. Oper. Res. Manag. Sci. 145, 269–293 (2010). https://doi.org/10.1007/978-1-4419-6142-6_7.
14. Codeca, L., Frank, R., Faye, S., Engel, T.: Luxembourg SUMO Traffic (LuST) Scenario: Traffic Demand Evaluation. IEEE Intell. Transp. Syst. Mag. 9, 52–63 (2017). https://doi.org/10.1109/MITS.2017.2666585.
15. Acosta, A.F., Espinosa, J.E., Espinosa, J.: Application of the SCRUM Software Methodology for Extending Simulation of Urban MObility (SUMO) Tools. In: Simulating Urban Traffic Scenarios. pp. 3–15. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-33616-9_1.
16. Dian Khumara, M.A., Fauziyyah, L., Kristalina, P.: Estimation of Urban Traffic State Using Simulation of Urban Mobility(SUMO) to Optimize Intelligent Transport System in Smart City. 2018 Int. Electron. Symp. Eng. Technol. Appl. IES-ETA 2018 - Proc. 163–169 (2019). https://doi.org/10.1109/ELECSYM.2018.8615508.
17. Bedogni, L., Gramaglia, M., Vesco, A., Fiore, M., Härri, J., Ferrero, F.: The Bologna ringway dataset: Improving road network conversion in SUMO and validating urban mobility via navigation services. IEEE Trans. Veh. Technol. 64, 5464–5476 (2015). https://doi.org/10.1109/TVT.2015.2475608.
18. Aditya, F., Nasution, S.M., Virgono, A.: Traffic Flow Prediction Using SUMO Application with K-Nearest Neighbor (KNN) Method. Int. J. Integr. Eng. 12, (2020).
19. HRIMECH, H., BENALLA, M., ACHCHAB, B., EL ALAOUI, A., EL HAIL, M.: Simulation optimization using SUMO: case of Casablanca. Int. J. Comput. Eng. Data Sci. 1, 28–36 (2021).
20. Alazzawi, S., Hummel, M., Kordt, P., Sickenberger, T., Wieseotte, C., Wohak, O.: Simulating the Impact of Shared, Autonomous Vehicles on Urban Mobility – a Case Study of Milan. Epic Ser. Eng. 2, 94–110 (2018). https://doi.org/10.29007/2N4H.
21. Golovnin, O.K., Pupynin, K. V., Privalov, A.S.: A Web-Oriented Approach for Urban Road Traffic Simulation. 2019 Int. Multi-Conference Ind. Eng. Mod. Technol. FarEastCon 2019. (2019). https://doi.org/10.1109/FAREASTCON.2019.8934302.
22. Maiorov, E.R., Ludan, I.R., Motta, J.D., Saprykin, O.N.: Developing a microscopic city model in SUMO simulation system. J. Phys. Conf. Ser. 1368, 042081 (2019). https://doi.org/10.1088/1742-6596/1368/4/042081.
23. Lu, H., Chen, J., Guo, L.: Energy Quality Management. In: Comprehensive Energy Systems. pp. 258–314. Elsevier (2018). https://doi.org/10.1016/B978-0-12-809597-3.00521-6.
24. Lindfield, G., Penny, J.: Particle Swarm Optimization Algorithms. In: Introduction to Nature-Inspired Optimization. pp. 49–68. Academic Press

(2017). https://doi.org/10.1016/B978-0-12-803636-5.00003-7.
25. Evers, G.I., Ghalia, M. Ben: Regrouping particle swarm optimization: A new global optimization algorithm with improved performance consistency across benchmarks. 2009 IEEE Int. Conf. Syst. Man Cybern. 3901–3908 (2009). https://doi.org/10.1109/ICSMC.2009.5346625.
26. Kennedy, J.: Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance. In: Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on. pp. 1931–1938 (1999). https://doi.org/10.1109/CEC.1999.785509.
27. Sun, J., Lai, C.H., Wu, X.J.: Particle Swarm Optimisation: Classical and Quantum Perspectives. CRC Press (2016).
28. Yang, X.-S.: Genetic Algorithms. Nature-Inspired Optim. Algorithms. 91–100 (2021). https://doi.org/10.1016/B978-0-12-821986-7.00013-5.
29. Rojas, M.H.G., Arellano, H.V., González, D.U., Rivera, M.M., Justo, M.O.A.: Steering Wheel Control in Lane Departure Warning System. Res. Comput. Sci. 2, 9–21 (2018).
30. Ramírez, C.A.M., Rivera, M.M.: Optimization of a Production Process from Demand and Number of Defective Units through Diffused Logic and Genetic Algorithms. Res. Comput. Sci. 2, 109–118 (2018).
31. Díaz-Nacar, E., Rodríguez-Vázquez, K.: Implementación en hardware de un algoritmo genético para resolver un problema combinatorio. Res. Comput. Sci. 8, 379–392 (2020).
32. Weise, T.: Global Optimization Algorithms - Theory And Application. (2009).
33. Weise, T.: Global optimization algorithms-theory and application. (2009).
34. Montes Rivera, M., Padilla Díaz, A., Ponce Gallegos, J.C., Canul-Reich, J., Ochoa Zezzatti, A., Meza de Luna, M.A.: Performance of Human Proposed Equations, Genetic Programming Equations, and Artificial Neural Networks in a Real-Time Color Labeling Assistant for the Colorblind. In: Martínez-Villaseñor, L., Batyrshin, I., and Marín-Hernández, A. (eds.) Advances in Soft Computing. pp. 557–578. Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-33749-0.