



Research Article /Review Article:

Bernstein Search Differential Evolution Approach for Real-time and Adaptive Traffic Signal Optimization in Urban Road Network

Abdullah Karaagac ^{*1}, Bulent Bostanci ²

¹ Department of Architecture and Urban Planning, Vocational School of Technical Sciences, Karamanoğlu Mehmetbey University, Karaman, Türkiye, e-mail: akaraagac@kmu.edu.tr

² Department of Geomatics Engineering, Erciyes University, Kayseri, Türkiye, e-mail: bbostanci@erciyes.edu.tr

* Correspondence: akaraagac@kmu.edu.tr

Article History:

Received: 09.08.2025
Revised 1: 08.10.2025
Revised 2: 14.10.2025
Published: 23.10.2025

Keywords:

Bernstein Search
Differential
Evolution
Algorithm;
Evolutionary
Optimization
Techniques;
Traffic Signal
Timing
Optimization;
Traffic Simulation.



This article is an open-access article distributed under the terms and conditions of the Creative Commons Attribution 4.0 International License (CC BY).

Publisher's Note:

This Journal remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Abstract

As urban populations surge, traffic congestion at multi-intersection networks becomes increasingly problematic, necessitating innovative solutions for traffic signal timing optimization. This study introduces the application of the Bernstein Search Differential Evolution (BSD) Algorithm, an evolutionary approach adapted specifically for traffic signal control optimization in a multi-intersection arterial. The BSD algorithm, notable for its parameter-free and adaptive nature, was applied to a complex urban traffic network in Kayseri, Türkiye. It was benchmarked against traditional algorithms such as the Genetic Algorithm (GA), Standard Differential Evolution (sDE), and the VISSIM software's algorithms. The number of vehicles entering the network in the morning, afternoon, and evening periods, which are the rush hours of the study area, has been obtained for 12 weeks. Through extensive simulations conducted in MATLAB and VISSIM, the BSD algorithm demonstrated superior performance, particularly during peak traffic hours, by significantly reducing vehicle wait times and improving traffic flow efficiency. Results indicated that the BSD algorithm reduced average vehicle wait times by up to 20% and increased traffic throughput by 15% compared to conventional methods. This study underscores the BSD algorithm's potential to significantly enhance traffic management in urban areas and suggests its broader application could effectively mitigate global urban traffic congestion.

Copyright © 2025 by the author(s).

Contribution to the field statement:

This study adapts the Bernstein Search Differential Evolution algorithm for traffic signal optimization across a four-intersection urban network. Using 12 weeks of traffic data and simulations, it evaluates performance against three established methods across vehicle throughput, queue lengths, and travel times. Results show improved overall performance, particularly during peak traffic periods.

Highlights:

- A modified evolutionary algorithm optimizes signals across four intersections.
- Twelve weeks of traffic counts inform simulations in MATLAB and VISSIM.
- Four optimization methods are compared across three performance criteria.
- The proposed method improves traffic flow, queues, and travel times.

Cite this article?

Karaagac, A., & Bostanci, B. (2025). Bernstein search differential evolution approach for real-time and adaptive traffic signal optimization in urban road network. *Advanced Geomatics*, 5(1), 8–26.

Özet

Kent nüfusunun artmasıyla birlikte, çok kavşaklı ağlardaki trafik sıkışıklığı giderek daha sorunlu hale gelmekte ve trafik sinyal zamanlama optimizasyonu için yenilikçi çözümler gerektirmektedir. Bu çalışma, çok kavşaklı bir ana arterde trafik sinyal optimizasyonu için özel olarak uyarlanmış evrimsel bir yaklaşım olan Bernstein Arama Diferansiyel Gelişim (BSD) Algoritmasının uygulamasını tanıtmaktadır. Parametre gerektirmeyen ve uyarlanabilir



yapısıyla dikkat çeken BSD algoritması, Türkiye'nin Kayseri şehrindeki karmaşık bir kentsel trafik ağına uygulanmıştır. Genetik Algoritma (GA), Standart Diferansiyel Gelişim Algoritması (sDE) ve VISSIM yazılımının algoritmaları gibi geleneksel algoritmalarla karşılaştırılmıştır. Çalışma alanının yoğun saatleri olan sabah, öğleden sonra ve akşam saatlerinde ağı giren araç sayısı 12 hafta boyunca elde edilmiştir. MATLAB ve VISSIM'de yapılan kapsamlı simülasyonlar aracılığıyla, BSD algoritması, özellikle yoğun trafik saatlerinde, araç bekleme sürelerini önemli ölçüde azaltarak ve trafik akış verimliliğini artırarak üstün performans göstermiştir. Sonuçlar, BSD algoritmasının geleneksel yöntemlere kıyasla ortalama araç bekleme sürelerini %20'ye kadar azalttığını ve trafik akışını %15 civarında artırdığını göstermiştir. Bu çalışma, BSD algoritmasının kentsel alanlarda trafik yönetimini önemli ölçüde iyileştirme potansiyelini vurgulamakta ve daha geniş çaplı uygulamasının küresel kentsel trafik sıkışıklığını etkili bir şekilde azaltabileceğini göstermeyi hedeflemektedir.

Anahtar Kelimeler: Bernstein Arama Diferansiyel Gelişim Algoritması, Evrimsel Optimizasyon Teknikleri, Trafik Sinyal Süresi Optimizasyonu, Traffic Simülasyonu.

1. Introduction

The rapid expansion of urban cities, accompanied by the surge in population numbers, has caused a remarkable increase in the volume of vehicles navigating the roads to facilitate daily commutes. This surge in vehicular presence compounds the already daunting issue of traffic congestion. The root cause of this urban traffic congestion is that the traffic demand far outstrips the existing capacity of urban traffic services during specific time intervals [1, 2].

Addressing the complexity of traffic congestion needs focused approaches to road intersections. Novel strategies for network-level intersection control modes are imperative to efficiently manage traffic flow in an urban traffic network, minimize delays, and enhance capacity and safety. These intersections hold substantial practical importance when tackling the ongoing challenges of traffic. Effectively mitigating traffic congestion at these intersections mandates a comprehensive and precise control of their service capabilities under diverse traffic scenarios by using traffic signal lights. This intricate issue can be framed as a problem of traffic signal control optimization (TSCO) that is persistent within urban traffic management [3].

Today, traffic networks have grown too complicated and cannot be managed manually. Thus, many different algorithms dedicated to TSCO have been developed for decades and tailored to address various applications, particularly in the context of the smart city paradigm.

Undoubtedly, fully autonomous vehicles represent the future of transportation. However, in the foreseeable future, it is likely that these vehicles will coexist with human-driven vehicles, sharing the same road infrastructure. In a completely automated environment, fully autonomous vehicles, due to their interconnectivity, may not require traditional traffic signals. Nevertheless, as long as human-driven vehicles are in operation, adherence to traffic signal systems will remain necessary, suggesting that the use of traffic signals will persist for at least another two decades. Consequently, the optimization of signal control systems retains its significance. Additionally, the advent of connected vehicles offers the potential to enhance signal timing optimization through their capability for real-time data sharing. Therefore, the development of diverse methodologies, including statistical analyses, artificial neural networks, and evolutionary algorithms, will continue to evolve and play a crucial role in the optimization process of traffic signal timing.

The randomness of the traffic demand is one of the pivotal sources of the uncertainty that causes the modeling challenges. Particularly at intersections, the uncertainty of traffic demand, the vehicle route choices, and the selection of lanes collectively cause the uncertainty, triggering vehicular queues and hysteresis phenomena. As a result, the uncertainty inherent in TSCO becomes closely tethered to the temporal delay experienced by vehicles at intersections. This delay hinges predominantly on traffic demand, road use mode, and the prevailing traffic control mechanisms at these junctions. It is imperative to acknowledge that ill-conceived traffic control measures at intersections can inadvertently exacerbate the already pressing issue of traffic congestion, adding another layer of complexity to the multifaceted challenge at hand [1, 3, 4].

Population-based metaheuristic algorithms allow for handling non-linear, continuous, and discrete optimization problems such as TSCO. They possess a distinctive characteristic that sets them apart: they operate without prior knowledge about the underlying characteristics of the search space. That provides simplicity when juxtaposed with deterministic methods necessitating using first-order and second-order information. The advantage of reduced complexity empowers metaheuristic algorithms to navigate the intricacies of complex optimization problems seamlessly. This attribute gains even more significance when dealing with large-scale engineering dilemmas, particularly those enveloped in uncertainties like the uncertainty of traffic flow [5].

Several prominent contenders emerge among the spectrum of evolutionary algorithms. Genetic Algorithms (GAs), Genetic Programming (GP), Evolutionary Strategies (ES), Evolutionary Programming (EP), and Differential Evolution (sDE) represent the vanguard. Each of these algorithms encapsulates a unique perspective on how to emulate the processes of evolution and natural selection to foster the refinement of solutions. These algorithms stand as a testament to the power of drawing inspiration from nature to tackle some of the most intricate challenges in optimization and engineering.



Despite the proliferation of evolutionary algorithms in traffic signal optimization, existing literature predominantly focuses on conventional methods like Genetic Algorithms (GA) and Differential Evolution (DE) without fully exploiting the capabilities of newer, more adaptive algorithms. Most studies concentrate on isolated intersections or non-complex traffic networks, thus limiting the applicability of their findings to the dynamic and often chaotic nature of urban multi-intersection traffic systems. Furthermore, while some algorithms have been adapted to real-time applications, they often require extensive parameter tuning or rely on predefined settings that do not adequately reflect the unpredictable patterns of urban traffic flows. There is a significant gap in research that employs advanced evolutionary algorithms, such as the Bernstein Search Differential (BSD) Algorithm, which are inherently parameter-free and offer a more flexible and robust approach to handling the complexities of urban traffic networks. This research aims to bridge this gap by demonstrating the efficacy of the BSD algorithm in optimizing traffic signals across multiple intersections in real-time, providing a substantial improvement in traffic management and efficiency.

This paper proposes an application of the Bernstein Search Differential Algorithm (BSD), a newly introduced evolutionary search algorithm, for multi-intersection traffic signal control optimization. The BSD algorithm is a handy, simple, structured, non-recursive, high-efficiency, fast, and practically universal parameter-free differential evolution algorithm. To scale and compare the performance of the BSD, the Genetic Algorithm (GA), the Standard Differential Evolution Algorithm (sDE), and the VISSIG Algorithm, which are frequently used in the literature, have also been applied. As the study area, one of the densest traffic networks, consisting of four intersections, has been selected in Kayseri city center, Türkiye. Implementing the algorithms directly on real traffic is impossible since the evolutionary process is iterative. For this reason, within the scope of the application, simulating the actual traffic count values for the proportion of vehicle numbers entering and leaving the network, queue lengths, and total travel times data are used to optimize the values of the traffic signals.

The statistics-based algorithm applied is the Vissig algorithm [6]. Vissig works with VISSIM software, which the PTV company developed. The values produced by this algorithm are currently used in the study area. VISSIM was used as a simulation environment during the application, and optimization algorithms were developed in MATLAB. The application is designed for an algorithm that works in real-time and offline. However, since evolutionary algorithms are based on iterations, it may take time to adapt to a pattern. It demonstrates an offline learning approach for rapid adaptation of the algorithm.

The contributions made to the literature by this study can be listed as follows.

- The BSD algorithm is not directly applicable to traffic due to its structure. In this study, a traffic-sensitive version of the BSD algorithm has been developed using its foundation.
- In order to be compatible with the traffic flow, the algorithm has been rearranged to approximate the density-flow graph, which is the basic diagram of traffic.
- A signal planning structure developed by the authors, which has not been used in any previous article, has been adapted. This allows the signal times estimated by the algorithm to be applied to the simulation environment.
- Optimization sub-parameters have been addressed specifically for this problem. In a classical approach, green and red signals are expected to be found by giving signal cycle times. However, in this application, a new approach has been demonstrated by using signal times, green signal times, and green signal start times. It is expected that this approach will inspire future studies.
- The developed algorithm has been tested on three generally accepted success criteria. These are the proportion of vehicles entering the network to vehicles leaving, total queuing at intersections, and total travel times. It is demonstrated that the modified BSD algorithm performs instant optimization in all three success criteria.

The rest of the paper has been organized as follows. The literature review discusses related works about using EAs in the TSCO problem. Section 3 defines the current problem, and the formulations are given. This section explains the TSCO problem's optimization process and the BSD proposed algorithm in detail. Section 4 solves an empirical example of an actual road network selected in Türkiye. Besides the modeling process, the application and results have been detailed. The limitation of simulation software and figuring out this problem with a signal scheduling algorithm has been discussed. Sections 5 and 6 share the conclusion of the entire paper and future studies.

2. Literature Review

The EAs have been widely used in traffic signal timing problems. Moreover, different approaches like the GA and sDE and their variants have been utilized in different traffic levels, single and multi-junctions, single or multi-objectives, with different inputs, adaptive and non-adaptive versions, and some with simulation software. This section examines the first studies, some critical milestones, and recent years' studies. The GA is the most common form of the EA; for this reason, the applications of the GA and its variants have been tackled first. Then, the applications of the sDE were considered due to the BSD algorithm being derived from the sDE.

The GA was first implemented by Foy et al. to optimize traffic signal timing. They tested the algorithm with four connected intersections and discussed the inputs, such as the number of cars and arrival volume. The study's objectives were optimizing the flow and minimizing average delay [7].

Sánchez-Medina et al. applied the GA to solve an over-saturated situation using hypothetical test cases in a large-scale actual case by using a cellular-automata-based micro-simulator, VISSIM. This study is one of the first



examples solved using the MATLAB-VISSIM combination. Four different objectives were tested separately: the number of vehicles (NoV) leaving the network, the mean travel time (MTT), the state of occupancy (SOC), and the global mean speed. The aim was to detect any correlation among the evolution of each variable. Finally, the authors discovered a robust linear relationship between the MTT and SOC and a slight relation between SOC and NoV. The work proposed using a multi-criterion objective function [2].

The GA was also used to reduce vehicle emissions. Zhang et al. proposed using the GA to combine microscopic level simulation and international vehicle emissions models to maximize the intersection capacity and reduce vehicle emissions. This multi-objective function was formulated as a single objective using weights and normalization to explain different units and saturation levels. The solution was formulated to calculate the green time of each intersection phase [8].

Chin et al. adopted the GA with integer encoding for TSOP, in which inputs are used as the offset, the cycle time, the phase sequence, and the green signal split. The purpose was to optimize the traffic delay and flow. The algorithm performance was evaluated in a two-junction network given two scenarios of undersaturated and oversaturated networks. The algorithm was used to avoid queue spillbacks [9].

Shen et al. applied the Non-dominated Sorting Genetic Algorithm (NSGA-II) [10] to tackle the TSOP with two objectives: average stop times and average delay time. They ran the algorithms in two scenarios, 4 and 18 intersections network, respectively, in Zhongguancun, China. The cycle times, the offset, and the green time of each phase were considered inputs [11]. Li et al. also examined the NSGA-II to optimize oversaturated cases. The objectives were to maximize the throughput and minimize the queue length. Simulations have been run on a 4-way intersection and utilized with signal timing optimization software Synchro [12].

Qian et al. presented an improved real-coded GA to optimize an actual intersection in Shenzhen, China. The GA optimized the traffic emission, average delay, and traffic capacity. Green time for each traffic light was optimized as output [13].

Dezani and Marranghello utilized the GA to optimize real-time traffic signal timings and adapt to find the best vehicle routes [14]. This study was an extension version of their other study [15], which optimized only vehicle routes. Experiments were run using a network of 12 intersections, and analysis illustrated that the adopted GA provided supply chain optimization by keeping vehicles on the road for the shortest time possible.

Tan et al. addressed the GA to minimize the vehicle delay under oversaturated conditions. The inputs were considered as the effective green time ratio in every phase. A single intersection simulation environment was designed based on the guidelines of the Public Works Department Malaysia (JKR). Simulations showed that GA could provide appropriate green time for each phase based on the traffic demand [16].

Yao et al. adopted the GA to develop a dynamic platoon dispersion model based on probe vehicle trajectory data to predict vehicle arrivals. They formulated a TSO model to optimize phase timings for vehicle delay minimization [17].

Xu et al. proposed the adaptive signal timing model by using GA. Due to the poor robustness of classical methods, they proposed an adaptive signal optimization model using vehicles' real-time position and speed information. The model aimed to optimize the minimum average vehicle delay as outputs and the green light duration of phase with limitations [18].

Yadav and Nuthong used a bi-level approach by combining the GA and Gradient Descent to improve the performance of the traffic signal timings. The aim was to minimize the waiting times of vehicles on the road. They used SUMO as simulation software [19].

Liu et al. developed it to manage intersection-based vehicle delay and parking times and propose an intersection signal control model with the minimum parking times and delay as the optimization objective using GA. They developed an optimization model for TSOP in a vehicle-road coordination environment, considering phase saturation as the index of the control effect and the influence of speed guidance on the vehicle's running state. They used MATLAB-VISSIM combination for development and simulation environment [20].

Wang et al. presented a GA approach for a multi-objective adaptive traffic signal control model based on multi-agent cooperation to optimize vehicle travel time and waiting time [21].

Bi et al. [22] have used sDE, another example of EAs. They optimized multi-agent Type-2 fuzzy logic control using the DE to achieve the best signal timings. Discrete Differential Evolution and Differential Evolution Bacteria Foraging Algorithms, the different variants of sDE, are good instances implemented for traffic signal optimization problems [5, 23].

Ceylan formulated an sDE design for a signal-controlled road network as a bi-level optimization problem. In lower levels, user route selections were used to determine traffic assignments that affected upper lever performance. The aim was to optimize total network performance. The optimization outputs were network cycle time, intersection green times, and offset variables [24].

Cakici and Murat utilized DE to adaptively optimize the signal timing and phasing of a 3-intersection network. DE was used to minimize the average vehicle delay. Decision variables were the three signal times of three phases. The proposed DE approach resulted in an average vehicle delay reduction in the range of 28%–42% and 3%–38% compared to the optimum fixed-time signal control and vehicle-actuated signal control for the studied test cases, respectively [25].



Baskan proposed a multi-objective bi-level model to optimize traffic signal timings in an environmentally friendly way. The upper level was modeled as a multi-objective parameter, maximizing the network capacity and minimizing vehicle emissions. The lower level was designed to evaluate the users' reactions. DE was applied at the higher level. DE was used for computation at both levels. The dedicated inputs were the O-D (Origin-Destination) multipliers, the cycle times, and the green times. The framework was applied to two networks with 2 and 9 signalized intersections, respectively [26].

Lin et al. utilized the fuzzy control method to sequence DE mutation operation in TSOP. They designed a multi-objective and adaptive signal control infrastructure using an average delay model, average queue length model, total delay calculation model for vehicles at intersections, and vehicle exhaust emission model as input parameters. The study mainly aimed to optimize the traffic capacity and minimize the delay rate [1].

3. Materials and Proposed Method

A primary concern in urban areas with complex traffic patterns is effectively controlling signal timings at intersections to minimize congestion, reduce travel times, and enhance overall traffic flow. This study addresses this challenge by optimizing real-time signal timings for a 4-junction urban main artery. To enhance the research in previous studies that lack the joint consideration of real-time and adaptive optimization approaches, this research proposes an improved algorithm to more efficiently achieve a quick-response, real-time, and adaptive optimization of signal timings. This research introduces a novel approach by reconsidering the optimization parameters to achieve real-time and adaptive optimization of signal timings in an urban network.

Ninety-four parameters are considered for rapid simultaneous optimization. These parameters include cycle durations for four signal controllers, green signal starting times for 45 signal groups, green signal durations for the 45 signal groups, and a fixed yellow signal duration of 3 seconds, as shown in Figure 1. The sum of green signal beginning times, green signal durations, and yellow signal durations determines the start time for the subsequent red signal phase.

The optimization takes into account the number of cars that have entered and left the network, the length of queues on each link, and the average times for the entire network. The resulting output values show the ratio of vehicles that enter and exit the network in the next step. These values have been generated based on the absolute count data of the study area.

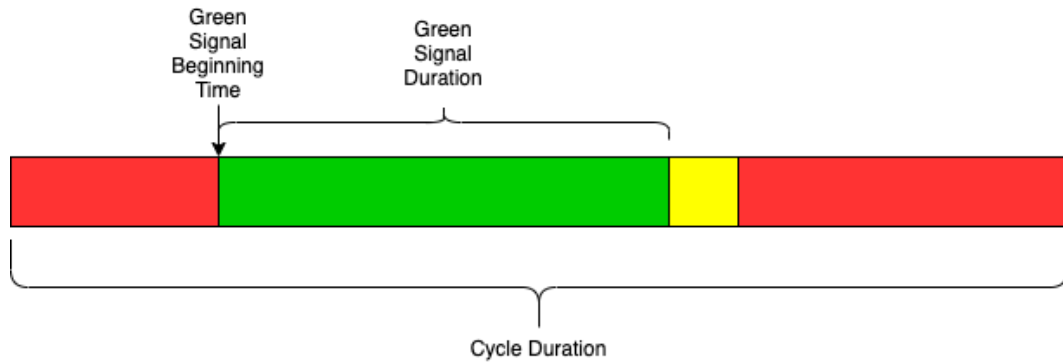


Figure 1. Optimization Sub-parameters.

On the other hand, optimization was designed according to three different success criteria as outputs:

1) Proportion of vehicles entering the network to vehicles leaving: Data were obtained with vehicle counter detectors placed at the entrances and exits of the network. This is considered the most valid method that can be applied in reality. The objective function of this method is shown in Eq.1.

$$F(N_{in}, N_{out}) = \sum_{i=1}^{1500} \left(\frac{N_{in}^{s(i)}}{N_{out}^{s(i)}} \right) \quad (1)$$

where $N_{in}^{s(i)}$ represents the number of vehicles entering the network and $N_{out}^{s(i)}$ represents the number of vehicles leaving.

2) Total queuing at intersections. Although travel time and queuing cannot be obtained precisely in real life, there are definitions. The queuing objective function for each link is shown in Eq. 2.

$$F(Q) = \underset{(Q)}{argmin} \sum_{i=1}^{1500} \sum_{q=1}^{79} \frac{N_r^{s(i)} + N_U - N_D - k_m(d(i) + W_A) + k_j W_A}{k_j - k_m} \quad (2)$$

where $N_r^{s(i)}$ represents the number of vehicles entering to link r , N_U and N_D are the number of vehicles passing through the upstream and downstream, respectively, k_m and k_j are critical density and the optimal density of the traffic flow, $d(i)$ is the actual distance between the upstream and downstream sections and W_A is the wave starting position.



3) Total travel times: the average total time of each vehicle's stay in the network. Travel times have been obtained directly from VISSIM.

Within the scope of the application, the average traffic information of 4 intersections observed for 12 weeks has been taken. The observations have been made between 07:00-09:00 and 16:00-18:00, during the rush hours of the day, and between 12:00-14:00, which is the hour of the average routine. Additionally, an oversaturated scenario has been created which never happens in routine to observe the algorithms' behavior. In the optimization application, periods of every 5 minutes have been optimized. It has been observed that hysteresis increased and success decreased because the vehicles could not adapt to the changing signal structure in shorter periods. Traffic hysteresis refers to a phenomenon in traffic flow dynamics where drivers' behavior does not immediately respond to changes in traffic conditions. It is a delay or lag in how traffic reacts to density, speed, or other changes.

The time sampling interval was used at 1 second. Although 1-second sampling intervals are not determinant for traffic, this sampling interval was necessary for correctly determining signal durations. Otherwise, the duration of the signal must also be multiples of selected seconds, which negatively affects success.

Evolutionary search algorithms could take random numbers as the initial value. However, they can make limited and unlimited searches. Limited search works between individuals' upper and lower limits [27–29].

For the optimum to converge to the solution faster and the algorithm to reach more vehicles to the outputs, some limitations needed to be added to the cycle duration and green signal duration (min. ten sec.) values. If lower and upper limits are not set, the algorithm only gives a green phase to certain links according to the type of objective function and ignores other links. Although this seems to increase the algorithm's success, it is an unacceptable approach in a real scenario. The most critical part of the limitations is that all produced values had to take positive integers. The upper and lower limits of the cycle durations for four intersections have been determined based on experience, as shown in Table 1.

Table 1. The Lower and Upper Determined for the Cycle Durations.

	K1	K2	K3	K4
Lower Bound (sec.)	90	90	80	80
Upper Bound (sec.)	120	120	100	100

Evolutionary search algorithms are based on the GA; the principles of selection, mutation, and crossover operators directly affect the success of the algorithms. For example, one of the three chromosomes selected from the initial population of the sDE leaves for selection. At the same time, the other two are used for mutation, making it extremely easy to converge to the global result in the search space. The proposed method BSD algorithm uses Bernstein polynomials in the sDE's crossover strategy, which speeds up solutions in global spaces. The reason for this is the uniform distribution of the crossover parameters used in the sDE and the GA while maintaining the homogeneity of the system.

The BSD algorithm was proposed by Çivicioğlu and Beşdok in 2019 [30]. It is an upgraded Standard Differential Evolution Algorithm (sDE) version. The main difference is that using Bernstein polynomials in the crossover process expedites convergence to optimal solutions in environments that fit curve infrastructure, just as urban traffic dynamics do.

The initial setup of the BSD technique involves assigning random values to its structural parameters during the initial phase. Furthermore, the creation of each pattern vector within the pattern matrix is conducted separately. Given that the advancement of each model vector occurs autonomously from the rest, it follows inherently that the BSD operates as a parallel search algorithm. The manipulation of the crossover phase is overseen using Bernstein Polynomials, as shown in Figure 2. The statistical background of the Bernstein Polynomials has been declared in Appendix a.

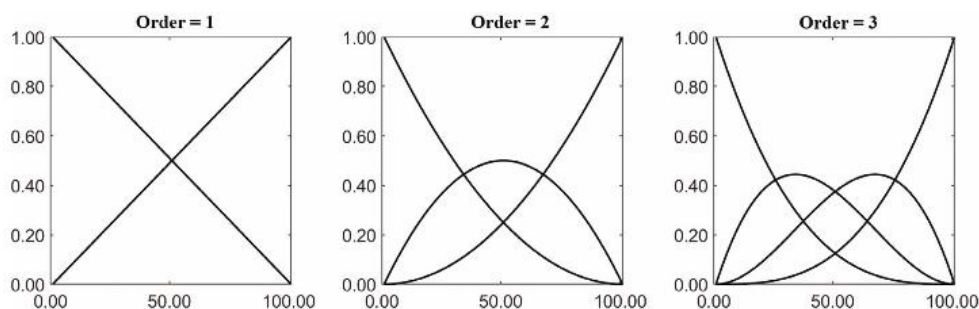


Figure 2. First, Second and Third-order Bernstein Polynomials.

Consequently, the BSD does not require a distinct parameter to regulate the crossover procedure. Like its evolutionary algorithm counterparts, the shared parameter values influence the efficacy of resolving problems using the BSD.



In a classical signal optimization process, preliminary parameters such as road geometries or adjacency matrices must be defined. However, the BSD algorithm can exhibit a geometry-independent numerical optimization approach.

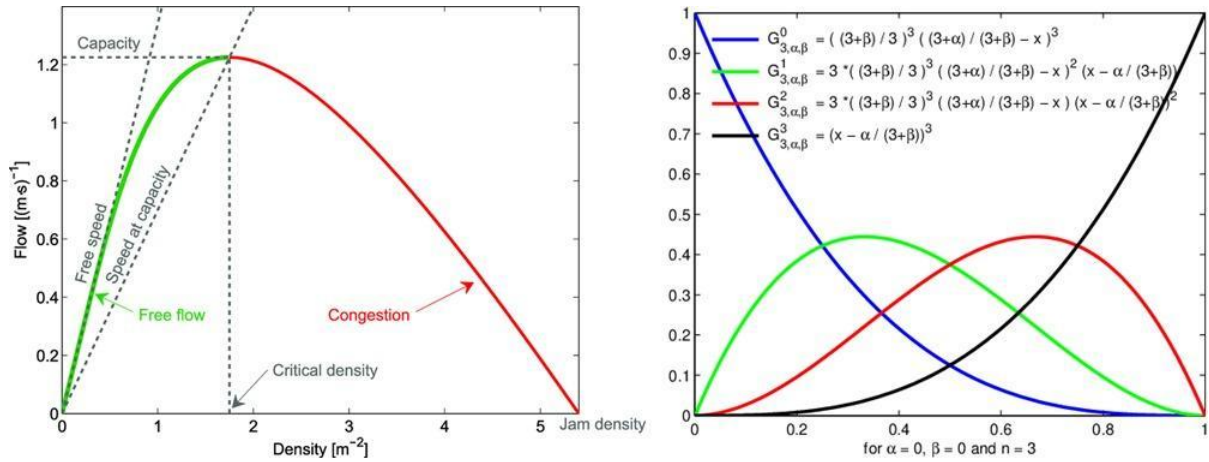


Figure 3. Similarity Flow-Density Diagram and Third-Order Bernstein Polynomials [31, 32].

The innovations provided by the BSD algorithm in signal timing are as follows:

- The BSD uses a unique bijective mutation strategy.
- The BSD is a preliminary parameter-free and geometry-independent approach.
- The BSD uses the Bernstein Polynomials as a mutation operator, which is more effective than sDE's crossover operators. Unlike sDE, BSD does not have a predetermined mutation and crossover value. Thus, the BSD algorithm can fit the traffic link's basic diagram, especially in higher complexity, as shown in Figure 3. The Mutation operator chooses a Bernstein Polynomial according to the traffic flow and density parameters. In lower complexity, when the traffic flow is free flow situations, 1st or 2nd-order Bernstein Polynomials are selected. When the traffic flow grows to congestion in higher complexity, 3rd-order Bernstein Polynomials are selected. In order to keep randomness, the mutation strategy selects the polynomials differently.
- Since BSD is a non-recursive method, it is easy to use BSD with parallel calculation methods.
- Unlike sDE and modern adaptations of sDE, BSD does not require parameter adjustment steps for the internal parameters of the mutation operator.

3.1. BSD Algorithm

In the BSD algorithm, chromosomes are called the pattern vector, and the population is called the pattern matrix. For this problem, the pattern matrix forms G pattern vectors as the green starting times and green durations and the pattern matrix L as the cycle length times, with a random solution. Each pattern vector consists of D individuals. For this problem, the BSD is designed as a global minimizer algorithm that performs limited searches.

The computation steps of BSD have been given below based on Discrete Mathematics symbology:

- 1- In the BSD, individuals are determined using Eq. 3 using the lower (low_j) and upper (up_j) limits of the cycle length time, as in Table 1.

$$L_{i,j} \sim U(low_j, up_j) | i = \{1: G\}, j = \{1: D\}, i, j \in Z^+ \quad (3)$$

where U , and Z^+ denotes *Continuous-Uniform Distribution* function and *Positive-Integers*, respectively.

- 2- The value of fitness function (F) of the pattern vectors ($fitL_i$) are calculated using Eq. 4:

$$fitL_i = F(L_i) \quad (4)$$

- 3- In the global minimizer pattern vector, the objective function value of the best solution to the problem ($bestP$) and the global minimizer pattern vector ($solP$) is obtained by Eq. 5.

$$[solL, bestL] = [fitL_{(\gamma)}, L_{(\gamma)}] | fitL_{(\gamma)} = (fitL) | \gamma \in \{1: G\} \quad (5)$$

where $fitL_{(\gamma)}$ denotes the objective function value of the pattern vector, $L_{(\gamma)}$.

- 4- The BSD controls the mutation rate with M by using Eq. 6 and Eq. 7. Here, the starting value of M is determined by Eq. 4 [30]

$$M_{(i=1:G, j=1:D)} = 0 \quad (6)$$

$$M_{(i, u(1: \lceil \rho \cdot D \rceil))} = 1 \quad (7)$$

where $\lceil \rho \cdot D \rceil$ operator rounds each element of $\rho \cdot D$ to the nearest integer greater than or equal to that element.

- 5- The ρ value here is determined by Eq. 8 using Bernstein Polynomials [33]. The 2nd order, Bernstein Polynomials, is explained in Appendix 1.



$$\begin{aligned}
 & \text{switch } \kappa_0 \\
 & \text{case 1 } \rho = (1 - \beta)^2 \\
 & \text{case 2 } \rho = 2 \cdot \beta \cdot (1 - \beta) \\
 & \text{case 3 } \rho = \beta^2 \\
 & \text{endsw}
 \end{aligned} \tag{8}$$

where $\beta \sim U(0,1)$ and $\kappa_0 = [3 \cdot \kappa_1^3]$, $\kappa_0 \sim U[0 \ 1]$, $\kappa_0 \in \{1: 3\}$ that are generated by using *Mersenne Twister* pseudo-random number generator method. The vector u in Eq. 5 is defined using Eq. 9:

$$u = \text{permute}(1: D) \tag{9}$$

where $\text{permute}(\cdot)$ denotes the changing process of the linear order of the vector, $(\cdot)$.

6- Evolutionary step range (F) is defined by using Eq. 10:

$$\{if \kappa_2 < \kappa_3, F = ([\eta_{(1,1:D)}^3 \circ |\lambda_{(1,1:D)}^3|] \times Q_{(1,1:G)})' \text{ else, } F = \lambda_{(G,1)}^3 \times Q_{(1,D)}\} \tag{10}$$

where κ_2 , κ_3 , η and λ are random values changes in every iteration according to the traffic pattern, where $\kappa_{2:3}$, $\eta \sim U(0,1)$, $\lambda \sim N(0,1)$ and $Q_{(m,n)} = 1$ which is $m \times n$ sized ones matrix. Here, “ $\circ$ ” operator denotes the *Hadamard Product* (i.e., *element-by-element* product).

7- Production processes of the BSD's trial pattern vector as green time durations (such as T_i) consist of random mutation processes. Trial pattern vectors in the BSD are produced using Eq. 11:

$$T = P + F \circ M \circ ((w^*)^3 \circ E + (1 - (w^*)^3) \circ \text{best}P - P) | w_{(1:G,1)}^* \sim U(0,1) \tag{11}$$

where $E = w \cdot P_{C_1} + (1 - w) \cdot P_{C_2} | w_{(1:G,1:D)} \sim U(0,1)$ and C_1 and C_2 are defined by Eq. 12.

$$C_1 = \text{permute}(1: G), C_2 = \text{permute}(1: G) | C_1 \neq [1: G], C_1 \neq C_2 \tag{12}$$

8- If an individual of any trial pattern vector exceeds the size of the search space of cycle lengths, the individual is reupdated using Eq. 13:

$$\begin{aligned}
 & \text{If } (T_{i,j} < \text{low}_j) \text{ or } (T_{i,j} > \text{up}_j) \\
 & \text{then } T_{i,j} = \text{low}_j + \delta \cdot (\text{up}_j - \text{low}_j)
 \end{aligned} \tag{13}$$

where, $\delta \sim U(0,1)$.

9- The objective function, $F(\cdot)$, and $\text{fit}T$ of the trial pattern vectors are calculated using Eq. 14.

$$\text{fit}T = F(T) \tag{14}$$

Here, the objective function changes according to the success criteria described before.

10- A trial pattern vector, is used to update the related pattern vector, which provides a better target function value than the corresponding pattern vector. The model is also updated with the objective function value of its vector. This is done using Eq. 15:

$$\text{If } \text{fit}T_{(i^*)} < \text{fit}P_{(i^*)}, [P_{(i^*)}, \text{fit}P_{(i^*)}] = [T_{(i^*)}, \text{fit}T_{(i^*)}] | i^* \in [1: G] \tag{15}$$

11- In the current iteration step, the best-solution model vector ($\text{best}P$) and the goal function value ($\text{sol}P$) are obtained using Eq. 16:

$$[\text{sol}L, \text{best}L] = [\text{fit}L_{(Y)}, L_{(Y)}] | \text{fit}L_{(Y)} = \min(\text{fit}L) \tag{16}$$

The flowchart of the BSD Algorithm is shown in Figure 4.

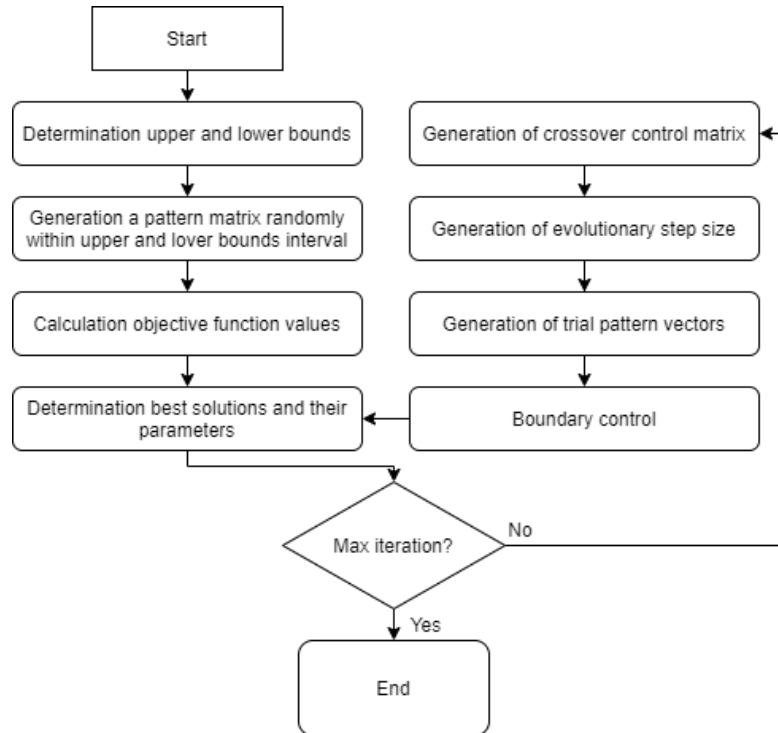


Figure 4. Flowchart of the Applied BSD Algorithm.

4. Application

4.1. Study Area

As the study area, 4 intersections in one of the densest urban traffic networks within the responsibility area of Kayseri Metropolitan Municipality, Türkiye, have been selected. Their vehicle counts have been made, and traffic volumes and turn rates at the intersections have been measured.

The locations of the 4 intersections measured have been illustrated on satellite image in Figure 5 as K1, K2, K3, and K4. Their critical saturation degrees are 83,27%, 84,67%, 88,49%, and 87,21%, respectively, considering 150 m length along from the signal heads. Also, the maximum velocity throughout the network, the average cruising speed, changes between 59 to 67 km/h. Current phase directions and signal times have been given in Figure 6. The Study Area is approximately 1 440 000 m^2 . The measurements were made between 07:00 - 09:00, 12:00-14:00, and 16:00-18:00 for 12 weeks. The sketches and vehicle count of each intersection have been given in Figure 7.



Figure 5. Satellite Image of Study Area.

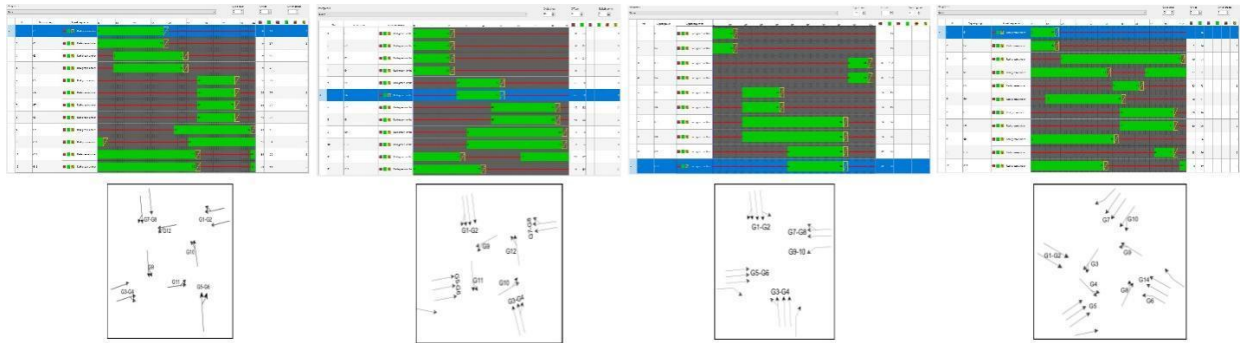


Figure 6. Current Signal Cycles and Phases Optimized by Vissig.

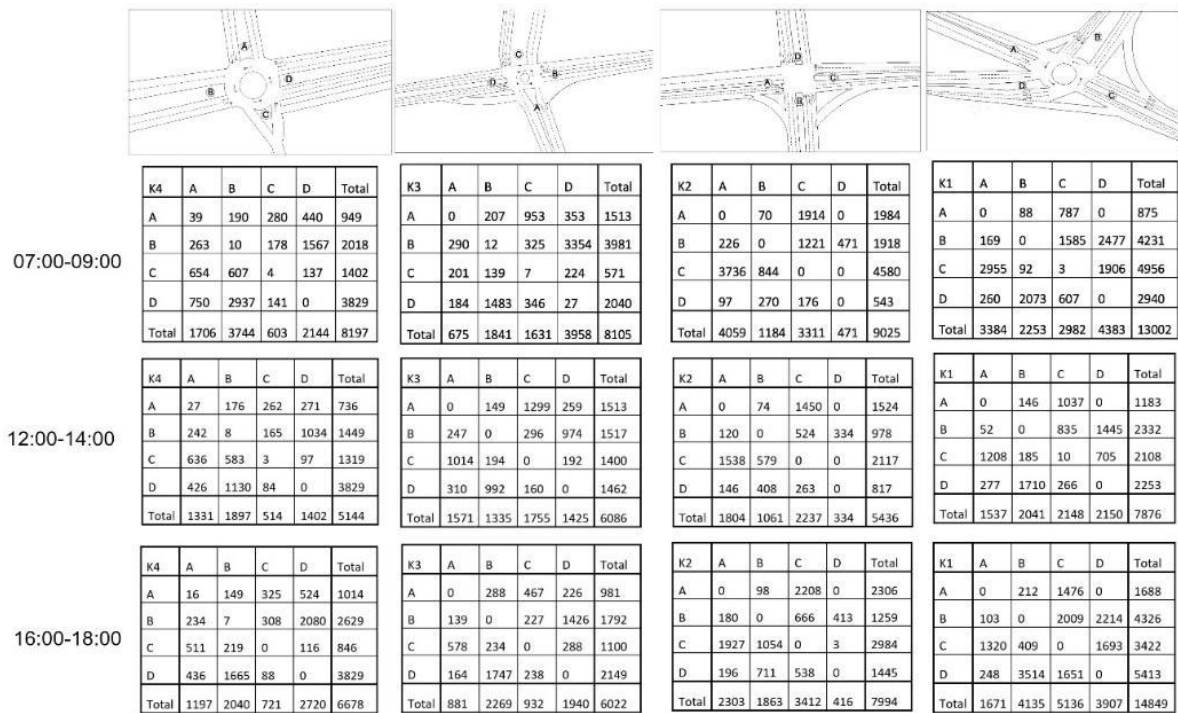


Figure 7. Intersection Sketches and Vehicle Counts.

4.2. Creating Simulation Scenarios and Application of Proposed Signal Scheduling Algorithm

The virtual model of the study area, in Figure 8, has been created in the VISSIM environment using a satellite image-based map. Figure 7 shows the real sketches and vehicle counts of the study area. These counts have been used as the system's input data. The intersection legs have been demonstrated as letters (A, B, C, D) in vehicle counts and marked in the sketches.

There are 4 intersections, 10 entrances, and 10 exits in the study area. There are 79 links on the network consisting of main links and hub links. Values such as length, width, number of lanes, and expected maximum speeds of each link had been modeled. Vehicle input values obtained from the measurements have been added to the link heads accessed to the network. The network has 1 signal controller for 4 intersections and 45 editable signal groups. A total of 20 vehicle counter detectors are placed in the inlet and outlet links. The relative flow values of the routes have been arranged manually. In this way, it has been ensured that the vehicles flow not to each link equally but to the busy regions, as in reality. In Figure 8, the stripes seen in the intersection areas are planned as speed reduction zones. Thus, when the vehicles arrive at the region, their signals decrease speed even if green. In addition, the transition advantage is observed in the right turns compared to the vehicle coming.

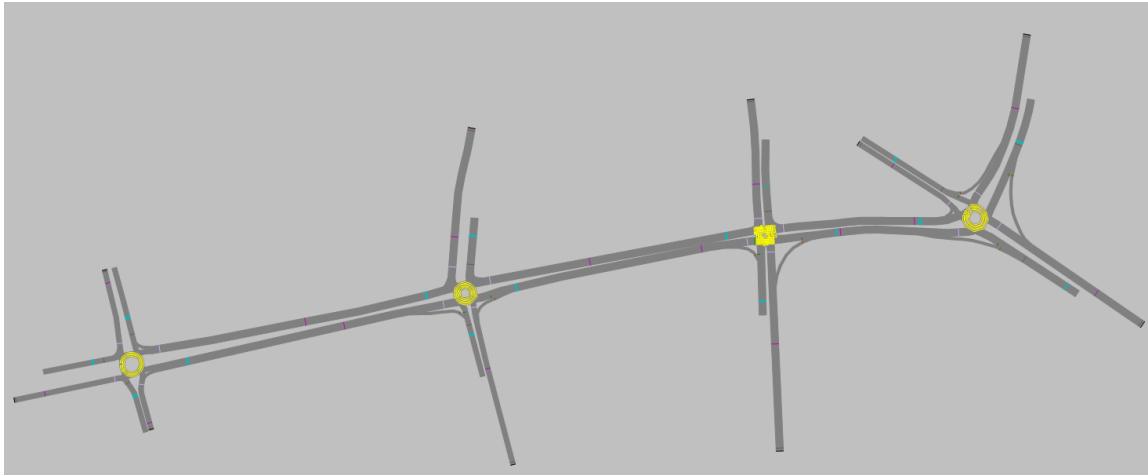


Figure 8. VISSIM model of the Study Area.

The method presented in this paper could work both in real-time and offline. However, since evolutionary algorithms are based on iterations, it may take time to adapt to a pattern. It demonstrates a pre-learned approach for rapid adaptation of the algorithm. This approach is trained with 100 different scenarios in an offline environment before a real-time online simulation. These scenarios were obtained using different random seeds. These seeds include highly congested and low-dense situations. The algorithm records these for a long time, and thanks to the GPU, it can adapt to each changing 5-minute segment in less than a second. Thus, responding to the changing follow-up level in a sufficient time is possible. However, as the workspace grows, the GPU requirement that needs to be used will increase.

For the application, each 5-minute part of the two-hour measurements has been individually optimized within itself. Since the duration was under this period, it has been observed that hysteresis had increased, and success had decreased because the vehicles could not adapt to the frequently changing signal structure. If the time is kept too long, the success of the signal optimization decreases as changes in traffic behavior have been observed. In other words, 5 minutes could be considered when the behavior changes at the micro level. The sampling interval has been used as 1 second. Although 1-second sampling intervals are not determinant for traffic, this sampling interval was necessary for correctly determining signal durations. Otherwise, the duration of the signal must also be multiples of selected seconds. This situation negatively affects success.

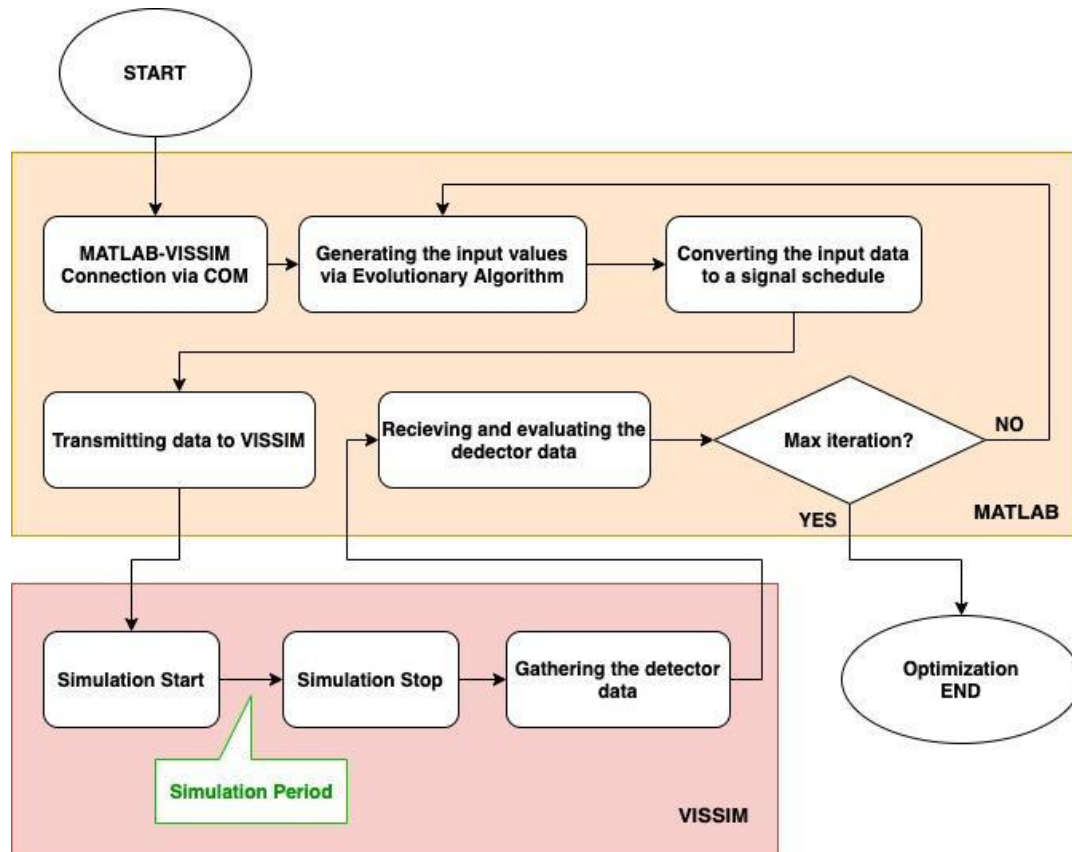


Figure 9. Flowchart of the Application.

The first criterion is the percentage of vehicles reaching the 10 exits in the traffic network compared to the vehicles entering. The Vissig algorithm is based on this criterion on the basis of intersections. The objective function of this method has been given before in Eq. 1, queuing has been shown before in Eq. 2, and travel time data are taken directly from the VISSIM program. In this study, minimizing the total instant queuing on all links was determined as another success criterion, as shown before in Eq. 2. As for the travel time, VISSIM calculates the travel time of each vehicle traveling from the entrances and exits determined during the simulation process and gives the average information. The sum of these averages is another success criterion in this study. Cars, light trucks, minibuses, and buses in the determined traffic network were counted and categorized as vehicles. On the other hand, heavy trucks are not allowed to enter this network. In the vehicle counts in the simulation processes, it is assumed that vehicles occupy 1 unit of space, pickup trucks and minibuses occupy 2 units of space, and buses occupy 3 units of space. The flowchart of the application is given in Figure 9.

VISSIM software allows the manipulation of only the current signal phase. However, turning the signals into green, yellow, or red phases causes the relevant signal group to stay in the same phase until the simulation ends. For the signal group to turn into any preferred signal for a certain period, it is necessary to intervene in the simulation at the beginning and end of the phase period. This is the most challenging problem in determining the sub-parameters. Although it is possible to intervene in the simulation at every step, this prolongs the simulation time. Because of this limitation, a scheduling algorithm has been developed to speed up the time and make the right interventions when necessary, thus creating a signal phase for optimization by a signal scheduling algorithm, as illustrated in Figure 1.

The algorithm can be examined in two parts: the stages of creating the scheduling and intervening in the VISSIM environment at the relevant time in the created schedule.

The steps to create the schedule are as follows:

- 1- The generated cycle duration, green signal beginning time, and green signal duration are obtained.
- 2- First, all signal heads are turned to the red phase.
- 3- The time obtained by summing the green signal beginning time and the durations is determined as the red phase conversion start time.
- 4- If the obtained red signal beginning time exceeds the cycle length, the excess time is shifted to the beginning of the cycle.
- 5- The total number of loops occurring at 4 intersections is calculated during the simulation.
- 6- The time to turn green and red phases of all signal groups in the loops to be formed is calculated.
- 7- Phase information of the relevant signal controller and the signal group is indexed as 1 if green and 2 if red.
- 8- The time information of all signals is converted into a matrix



9- This matrix is sorted in rows from smallest to most significant according to the time information.

10- As a result, a scheduling matrix shows the time and phase information.

An example of the scheduling matrix is given in Table 2.

Table 2. Example of Signal Scheduling Matrix.

Interference Time	Signal Controller No.	Signal Group No.	Phase
21	2	2	1
53	1	9	2
60	1	11	2
72	4	5	2
77	3	6	2
82	1	8	1
88	4	9	1
91	1	5	1
91	2	3	2
98	2	1	2
104	1	11	1
104	2	3	1
105	1	8	2
105	2	1	1
110	1	3	1
111	1	2	1
117	1	3	2
118	1	7	2

In Table 2, the first column of the matrix indicates the intervention time, the second column indicates the signal controller number, the third column indicates the signal group number and the fourth column indicates the phase type. If the fourth column value in the matrix is 1, the phase turns green, and if it is 2, it turns red. For example, for the first row seen in Table 2, at the 21st simulation step (in seconds), the signals in signal group 2 (Leg B) at signal controller 2 (Intersection K2) will be turned green (because code 1 is green). In these simulations, the yellow phase lights up for 3 seconds before the red phase starts.

The following steps are followed in order to use the generated chart matrix in the VISSIM environment:

- 1- The simulation starts.
- 2- The simulation continues until the time in the matrix's first row and first column.
- 3- When it reaches the relevant time, the simulation is paused.
- 4- The second, third, and fourth columns are checked in the line where there is a pause.
- 5- The signal group of the third column of the signal controller in the second column is converted to the phase value in the fourth column.
- 6- The simulation continues.
- 7- If the simulation time is reached, the simulation stops. Otherwise, go to Step 2 and apply the same operations for the following line.
- 8- When the simulation stops, detector information is collected.
- 9- The ratio of vehicles entering the network and reaching the exit is calculated from the detector data. Travel times are gathered, and average travel times are calculated. Total queuing data is gathered during the simulation time.

5. Findings

The applications have been run 300 times (100 for each performance criterion) with 1500 iterations. Each simulation has been applied in each iteration by keeping the random seed value constant during each 5-minute interval to evaluate the results correctly. Then, random seed values have been changed for the next run. With this, 100 different scenarios have been applied, and their mean values have been accepted as the actual performance. In terms of performance, the proportions of vehicles that reached the exits, total queueing, and average travel times have been considered.

The graphics given in Figure 10- 12 show the iteration results of any selected simulation and the results of 100 simulations. As seen from the results, if the limitations have been given correctly, the performance of the BSD algorithm has surpassed others, especially in rush hours. For the Vissig algorithm, it is not possible to obtain iteration data. The result could be displayed directly. Therefore, the performance of the Vissig algorithm has been indicated with a purple line.

As seen in Figure 10, it can be said that the BSD algorithm is superior to the Vissig algorithm. As shown in Figure 10, although the Vissig algorithm gave a more successful result on average between 16:00 - 18:00, the results are remarkably close to each other. On the other hand, the apparent superiority of BSD comes to the fore in

oversaturated simulations. It is seen that sDE and GA, which are other evolutionary algorithms, lag behind BSD and Vissig in terms of the percentage of vehicles reaching the exits.

Based on total average travel times, the algorithm performances are close, especially in the less peak hours, 12:00 - 14:00. Here, the performance of the GA has increased, although it is in last place. It has been observed that its performance is ahead of sDE in the 16:00 – 18:00 range. In terms of total average travel times, BSD has a clear advantage. However, it can be said that he performed very closely with Vissig in oversaturated situations.

Based on queuing, BSD provides successful results in real traffic data. In the oversaturated state, Vissig has the advantage. Still, the difference in success is around 1%.

In conclusion, BSD has the best overall performance. The robust performance of the algorithms for 100 different scenarios shows that this performance is around 10%. The basis of this success is correctly determining the search space, which is the lower and upper bounds.

Although the basis of evolutionary search algorithms is based on the GA, the principles of selection, mutation, and crossover operators directly affect the success of the algorithms. For example, one of the three chromosomes selected from the initial population of the sDE leaves for selection. At the same time, the other two are used for mutation, making it extremely easy to converge to the global result in the search space. The BSD algorithm uses Bernstein Polynomials in the sDE's crossover strategy, which speeds up solutions in global spaces. The reason for this is the uniform distribution of the crossover parameters used in the sDE and the GA while maintaining the homogeneity of the system. On the other hand, the Bernstein polynomials eliminate this homogeneity and increase the diversity in the search space. Besides, the Bernstein polynomials are similar to the fundamental traffic flow diagram. As a result, it has been demonstrated that the BSD algorithm shows the best performance.

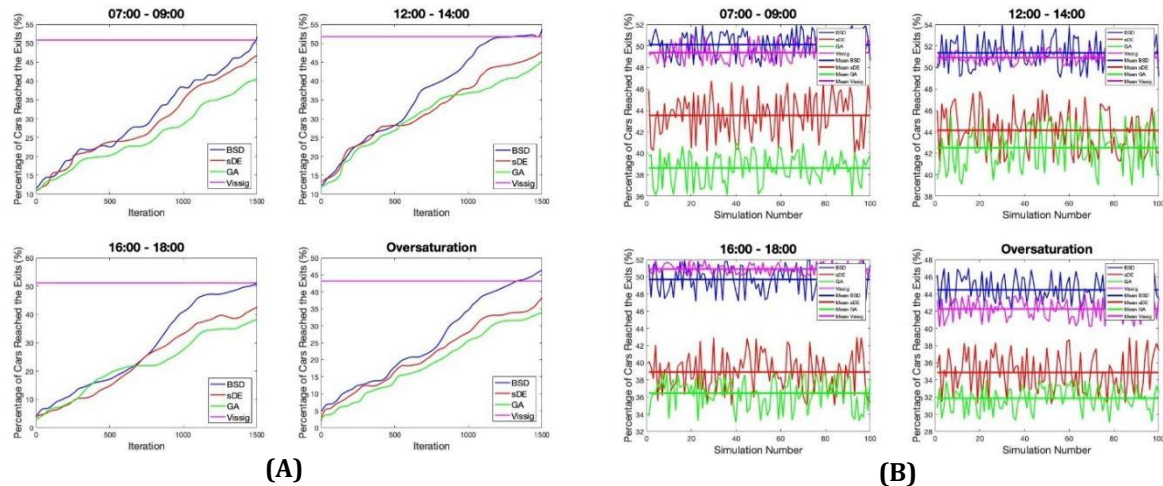


Figure 10. (A) An example for 1 of 100 simulations performance for Percentage of Cars Reached the Exits (B) Ratio Performances of Algorithms in 100 Different Scenarios.

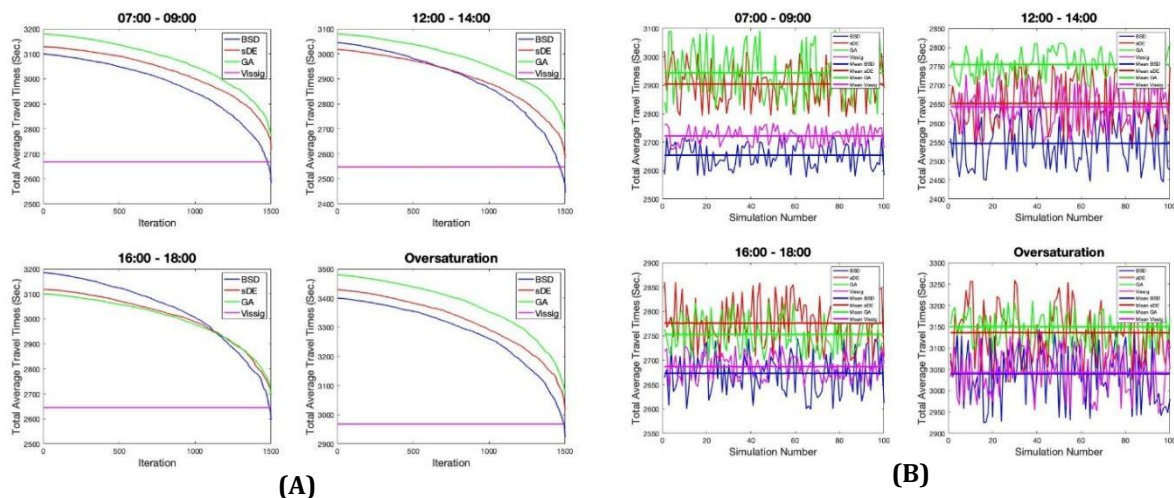


Figure 11. (A) An example for 1 of 100 simulations for Total Average Travel Times, (B) Total Average Travel Times Performances of Algorithms 100 Different Scenarios.

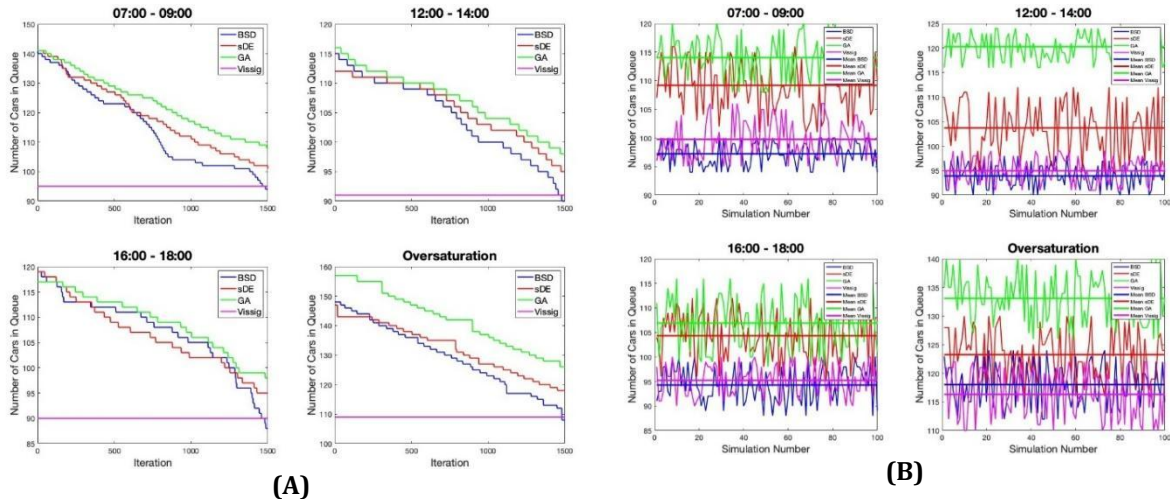


Figure 12. (A) An example of 1 of 100 simulations for Total Number of in Queue, (B) Number of Cars in Queue Performances of Algorithms 100 Different Scenarios.

6. Result and Discussion

This paper focuses on solving traffic signal timing problems via three evolutionary and one statistics-based algorithm. The study aims to prove and demonstrate the newly introduced evolutionary search algorithm, the BSD algorithm. The BSD algorithm uses the Bernstein polynomials during the crossover and mutation process, and they show similarity with the fundamental traffic flow diagram. Since GA and sDE, the other evolutionary algorithms used, are frequently encountered methods in the literature, they have been added to the application to scale the success of BSD. The statistics-based algorithm that was applied is the Vissig algorithm. Vissig works with VISSIM software, which was developed by PTV company. The values produced by the Vissig algorithm are currently used in the study area in real life.

An urban network of four intersections has been selected as the study area. The average traffic information of four intersections observed for 12 weeks has been taken. The observations have been made between 07:00-09:00 and 16:00-18:00, during the rush hours of the day, and between 12:00-14:00, which is the hour of the average routine. In the optimization application, periods of every 5 minutes have been optimized. It has been observed that hysteresis increased in case the optimizations were made in shorter periods. VISSIM was used as a simulation environment during the application, and optimization algorithms were developed in MATLAB.

Within the scope of the application, 94 sub-parameters have been optimized: cycle durations for four signal controllers, green signal starting times for 45 signal groups, and 45 green signal durations. The output performance value, the ratio of the total number of vehicles entering and leaving the network, the total average travel times of each route, and the total queue length of each link are considered.

The applications have been run 100 times on average with 1500 iterations. Each simulation was applied in each iteration by keeping the random seed value constant during each 5-minute interval to evaluate the results correctly. Then, random seed values have been changed for the next run. As a result of this, 100 different scenarios have been applied, and their mean values have been accepted as the actual performance.

As clearly seen from the results, in case the limitations have been given correctly, the performance of the BSD algorithm has surpassed about 1.1% against the performance of the VISSIG algorithm regarding the number of vehicles, this rate has become approximately 1.3% when they have been evaluated with normalized values. It has been seen that the performance of the sDE and the GA in point vehicles is about 6.3% and 7.2% compared to the BSD, whereas the decrease in their normalized performances are 11.6% and 13.5% respectively. It has been observed that the BSD adapts to the conditions more successfully. Accordingly, it has been indicated that the BSD shows better performance approximately 2.8% and 5.7% in normalized values in comparison with VISSIG. On the other hand, it has been observed that the performance values of the sDE and the GA showed lower performance approximately 6.7% and 8.42%, respectively, about 13.8% and 17.2% in normalized comparison against the BSD. Statistical results have been shown in Table 3-5.

Table 3. Performance Comparison of Total Average Travel Times in Different Time Intervals.

Total Average Travel Times (Sec.)				
07:00 - 09:00	BSD	GA	sDE	Vissig
Number of Simulations	100	100	100	100
Mean	2653	2943	2947	2722
Standard Deviation	255	273	271	261
Total Average Travel Times (Sec.)				
12:00 - 14:00	BSD	GA	sDE	Vissig
Number of Simulations	100	100	100	100



Mean	2547	2751	2652	2643
Standard Deviation	278	283	301	302
Total Average Travel Times (Sec.)				
16:00 - 18:00	BSD	GA	sDE	Vissig
Number of Simulations	100	100	100	100
Mean	2673	2753	2686	2737
Standard Deviation	276	308	271	284
Total Average Travel Times (Sec.)				
Over Saturation	BSD	GA	sDE	Vissig
Number of Simulations	100	100	100	100
Mean	3039	3150	3136	3042
Standard Deviation	276	308	271	284

Table 4. Performance Comparison of Percentage of Cars Reached the Exits in Different Time Intervals.

Percentage of Cars Reached the Exits (%)				
07:00 - 09:00	BSD	GA	sDE	Vissig
Number of Simulations	100	100	100	100
Mean	50,12	38,63	43,55	49,38
Standard Deviation	1,22	1,92	1,37	1,26
Percentage of Cars Reached the Exits (%)				
12:00 - 14:00	BSD	GA	sDE	Vissig
Number of Simulations	100	100	100	100
Mean	51,36	42,53	44,15	50,9
Standard Deviation	1,48	2,01	1,92	1,85
Percentage of Cars Reached the Exits (%)				
16:00 - 18:00	BSD	GA	sDE	Vissig
Number of Simulations	100	100	100	100
Mean	49,69	36,46	38,92	50,89
Standard Deviation	1,54	5,17	1,76	1,12
Percentage of Cars Reached the Exits (%)				
Over Saturation	BSD	GA	sDE	Vissig
Number of Simulations	100	100	100	100
Mean	44,48	31,85	34,86	42,25
Standard Deviation	1,49	2,23	1,38	1,26

Table 5. Performance Comparison of Total Number of Cars in Queue in Different Time Intervals.

Total Number of Cars in Queue				
07:00 - 09:00	BSD	GA	sDE	Vissig
Number of Simulations	100	100	100	100
Mean	97	109	114	99
Standard Deviation	2,04	4,4	3,45	2,81
Total Number of Cars in Queue				
12:00 - 14:00	BSD	GA	sDE	Vissig
Number of Simulations	100	100	100	100
Mean	93	120	103	95
Standard Deviation	2,48	5,32	5,62	2,62
Total Number of Cars in Queue				
16:00 - 18:00	BSD	GA	sDE	Vissig
Number of Simulations	100	100	100	100
Mean	94,28	106,9	104,32	95,24
Standard Deviation	3,51	4,84	5,3	4,01
Total Number of Cars in Queue				
Over Saturation	BSD	GA	sDE	Vissig
Number of Simulations	100	100	100	100
Mean	94,28	106,9	104,32	95,24
Standard Deviation	3,56	4,24	4,18	3,86

As seen from the results, if the limitations have been given correctly, the performance of the BSD algorithm has surpassed others, especially during rush hours. It has been observed that the performances were close to each other at the routine hours 12:00 to 14:00. Besides, it has been observed that the BSD algorithm finds a breaking point and converges rapidly to the result, especially during rush hours and other evolutionary algorithms could



converge to a similar result in some iterations eventually. As a result, it has been demonstrated that the BSD algorithm shows the best performance.

This application has been completed as an experiment using actual traffic data and environment. This allows it to be easily applied to other networks, especially by local authorities. Evolutionary algorithms have the excellent computing advantage of their easy editable structure and are compatible with other algorithms, such as reinforcement learning and supervised learning. Using Bernstein Polynomials is innovative and could make a significant breakthrough for future studies.

7. Conclusion

This study introduced and validated the effectiveness of the Bernstein Search Differential (BSD) Algorithm for optimizing traffic signal control in a multi-intersection urban traffic network. Our approach leverages the innovative application of evolutionary algorithms to address the persistent challenge of urban traffic congestion, specifically through the adaptation and employment of the BSD algorithm. Compared with traditional algorithms such as the Genetic Algorithm (GA), Standard Differential Evolution (sDE), and the algorithms embedded within VISSIM, the BSD algorithm demonstrated significant improvements in traffic flow efficiency and reduction in vehicle wait times during peak hours.

The statistical analyses underscored the BSD algorithm's superior performance, providing a robust framework for future implementations and research. The results indicate a potential reduction in average vehicle wait times by up to 20% and an increase in traffic throughput by 15% compared to existing methods. These findings highlight the BSD algorithm's capacity to adapt to real-time changes and optimize traffic flow, which are critical in managing the increasing complexity and volume of urban traffic networks.

Furthermore, the successful application of the BSD algorithm in the context of Kayseri, Türkiye, suggests its broader applicability to other urban environments experiencing similar challenges. Future studies could explore the integration of this algorithm with other intelligent transportation systems and technologies, enhancing its adaptability and efficiency across different urban landscapes.

Appendix A. Bernstein Polynomials

Sergei Natanovich Bernstein developed Bernstein polynomials for Curve Fitting problems in the numerical analysis area. Bernstein polynomials were developed initially to prove the Stone–Weierstrass approach, and over time, they became more critical than Bezier curves in space $[0, 1]$. These polynomials are essential in engineering and mathematical applications, especially for solving differential equations and integrals [34, 35]. n^{th} order Bernstein polynomials are defined by Eq. 17 [34]:

$$B_{s,n}(t) = \binom{n}{s} \cdot t^s \cdot (1-t)^{n-s}, 0 \leq s \leq n \quad (17)$$

Where $s = 0: n, \binom{n}{s} = \frac{n!}{s!(n-s)!}$

Eq. 17 could be generalized as Eq. 18:

$$B_{s,n}(t) = \binom{n}{s} \cdot t^s \cdot (1-t)^{n-s} = \binom{n}{s} t^s \left(\sum_{k=0}^{n-s} (-1)^k \binom{n-s}{k} t^k \right) = \sum_{k=0}^{n-s} (-1)^k \binom{n}{s} \binom{n-s}{k} t^{s+k}; j = 1, 2, 3 \dots n. \quad (18)$$

Bernstein Polynomials could be described as matrix form following:

$$\theta(t) = A \cdot T_n \cdot t \quad (19)$$

where $\theta(t) = [B_{0,n}(t), B_{1,n}(t), \dots, B_{n,n}(t)]^T, T_n = [1, t, \dots, t^n]^T$ and A is a matrix, shown in Eq. 20:

$$A = \begin{bmatrix} (-1)^0 \binom{n}{0} : 0 : 0 & (-1)^1 \binom{n}{0} \binom{n-0}{1-0} : (-1)^i \binom{n}{i} \binom{n-i}{1-i} : \dots & \dots : \dots : 0 & (-1)^{n-0} \binom{n}{0} \binom{n-0}{n-0} : \\ (-1)^{n-i} \binom{n}{i} \binom{n-i}{n-i} : (-1)^n \binom{n}{n} \end{bmatrix} \quad (20)$$

Eq. 21 produces $(n+1)$ sized n^{th} order Bernstein Polynomials. For each $s < 0$ and $s > n, B_{s,n} = 0$.

$$\begin{aligned} B_{0,2}(t) &= (1-t)^2 \\ B_{1,2}(t) &= 2t \cdot (1-t) \\ B_{2,2}(t) &= t^2 \end{aligned} \quad (21)$$

Where $B_{m,n}$ represents Bernstein Polynomials and interval of solution space t .

Acknowledgements

We thank Kayseri Metropolitan Municipality for supporting data sharing. This article is based on part of the Ph.D. thesis of Abdullah KARAAĞAÇ named "Traffic signal optimization via heuristic search algorithms".

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors. All data was gathered from Kayseri Metropolitan Municipality free of charge.



Conflicts of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data Availability Statement

All data used is shown on the paper in Figure 7. All data was gathered from Kayseri Metropolitan Municipality free of charge. This data is not publicly available; however, they share it in demand, especially for academic purposes.

Institutional Review Board Statement

Not Applicable

CRedit Author Statement

Abdullah Karaağaç: Conceptualization, Methodology, Investigation, Formal analysis, Visualization, Writing – original draft.

Bülent Bostancı: Supervision, Writing – review & editing.

Ahmet Yılmaz: Supervision, Methodology, Writing – review & editing, Project administration.

References

- [1] Lin, H., Han, Y., Cai, W., & Jin, B. (2023). Traffic signal optimization based on fuzzy control and differential evolution algorithm. *IEEE Transactions on Intelligent Transportation Systems*, 24, 8555–8566. <https://doi.org/10.1109/TITS.2022.3195221>
- [2] Sanchez-Medina, J. J., Galan-Moreno, M. J., & Rubio-Royo, E. (2010). 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*, 11, 132–141. <https://doi.org/10.1109/TITS.2009.2034383>
- [3] Fu, X., Gao, H., Cai, H., Wang, Z., & Chen, W. (2021). How to improve urban intelligent traffic? A case study using traffic signal timing optimization model based on swarm intelligence algorithm. *Sensors*, 21, Article 2631. <https://doi.org/10.3390/s21082631>
- [4] Zhao et al. (2021). [The bibliographic details of this source are missing in the submitted manuscript and must be completed by the authors.]
- [5] Liu, Q., & Xu, J. (2012). Traffic signal timing optimization for isolated intersections based on differential evolution bacteria foraging algorithm. *Procedia - Social and Behavioral Sciences*, 43, 210–215. <https://doi.org/10.1016/j.sbspro.2012.04.093>
- [6] PTV. (2018). [The bibliographic details of this source are missing in the submitted manuscript and must be completed by the authors.]
- [7] Foy, M. D., Benekohal, R. F., & Goldberg, D. E. (1992). Signal timing determination using genetic algorithms. *Transportation Research Record*, 108–115.
- [8] Zhang, B., Shang, L., & Chen, D. (2010). Traffic intersection signal-planning multi-object optimization based on genetic algorithm. In *2010 2nd International Workshop on Intelligent Systems and Applications (ISA)* (pp. 1–4). IEEE. <https://doi.org/10.1109/IWISA.2010.5473294>
- [9] Chin, Y. K., Yong, K. C., Bolong, N., Yang, S. S., & Teo, K. T. K. (2011). Multiple intersections traffic signal timing optimization with genetic algorithm. In *2011 IEEE International Conference on Control System, Computing and Engineering (ICCSCE)* (pp. 454–459). IEEE. <https://doi.org/10.1109/ICCSCE.2011.6190569>
- [10] Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6, 182–197. <https://doi.org/10.1109/4235.996017>
- [11] Shen, Z., Wang, K., & Wang, F.-Y. (2013). GPU based non-dominated sorting genetic algorithm-II for multi-objective traffic light signaling optimization with agent based modeling. In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)* (pp. 1840–1845). IEEE. <https://doi.org/10.1109/ITSC.2013.6728496>
- [12] Li, Y., Yu, L., Tao, S., & Chen, K. (2013). Multi-objective optimization of traffic signal timing for oversaturated intersection. *Mathematical Problems in Engineering*, 2013, 1–9. <https://doi.org/10.1155/2013/182643>
- [13] Qian, R., Lun, Z., Wenchen, Y., & Meng, Z. (2013). A traffic emission-saving signal timing model for urban isolated intersections. *Procedia - Social and Behavioral Sciences*, 96, 2404–2413. <https://doi.org/10.1016/j.sbspro.2013.08.269>
- [14] Dezani, H., Marranghello, N., & Damiani, F. (2014). Genetic algorithm-based traffic lights timing optimization and routes definition using Petri net model of urban traffic flow. *IFAC Proceedings Volumes*, 47, 11326–11331. <https://doi.org/10.3182/20140824-6-ZA-1003.01321>
- [15] Dezani, H., Bassi, R. D. S., Marranghello, N., Gomes, L., Damiani, F., & Nunes Da Silva, I. (2014). Optimizing urban traffic flow using genetic algorithm with Petri net analysis as fitness function. *Neurocomputing*, 124, 162–167. <https://doi.org/10.1016/j.neucom.2013.07.015>



- [16] Tan, M. K., Chuo, H. S. E., Chin, R. K. Y., Yeo, K. B., & Teo, K. T. K. (2016). Optimization of urban traffic network signalization using genetic algorithm. In *2016 IEEE Conference on Open Systems (ICOS)* (pp. 87–92). IEEE. <https://doi.org/10.1109/ICOS.2016.7881994>
- [17] Yao, Z., Jiang, Y., Zhao, B., Luo, X., & Peng, B. (2020). A dynamic optimization method for adaptive signal control in a connected vehicle environment. *Journal of Intelligent Transportation Systems*, 24, 184–200. <https://doi.org/10.1080/15472450.2019.1643723>
- [18] Xu, M., An, K., Vu, L. H., Ye, Z., Feng, J., & Chen, E. (2019). Optimizing multi-agent based urban traffic signal control system. *Journal of Intelligent Transportation Systems*, 23, 357–369. <https://doi.org/10.1080/15472450.2018.1501273>
- [19] Yadav, A., & Nuthong, C. (2020). Traffic signal timings optimization based on genetic algorithm and gradient descent. In *2020 5th International Conference on Computer and Communication Systems (ICCCS)* (pp. 670–674). IEEE. <https://doi.org/10.1109/ICCCS49078.2020.9118450>
- [20] Liu, S., Li, Y., Qiu, Y., Zhang, B., Qiu, S., & Liu, X. (2020). Signal timing optimization algorithm for an intersection connected with an urban expressway. *Arabian Journal for Science and Engineering*, 45, 8667–8684. <https://doi.org/10.1007/s13369-020-04835-6>
- [21] Wang, H., Hu, P., & Wang, H. (2021). A genetic timing scheduling model for urban traffic signal control. *Information Sciences*, 576, 475–483. <https://doi.org/10.1016/j.ins.2021.06.082>
- [22] Bi, Y., Srinivasan, D., Lu, X., Sun, Z., & Zenf, W. (2014). Type-2 fuzzy multi-intersection traffic signal control with differential evolution optimization. *Expert Systems with Applications*, 41(16), 7338–7349. <https://doi.org/10.1016/j.eswa.2014.06.022>
- [23] Zhang, M., Zhao, S., Lv, J., & Qian, Y. (2009). Multi-phase urban traffic signal real-time control with multi-objective discrete differential evolution. In *2009 International Conference on Electronic Computer Technology (ICECT)* (pp. 296–300). IEEE. <https://doi.org/10.1109/ICECT.2009.38>
- [24] Ceylan, H. (2013). Optimal design of signal controlled road networks using differential evolution optimization algorithm. *Mathematical Problems in Engineering*, 2013, 1–11. <https://doi.org/10.1155/2013/696374>
- [25] Cakici, Z., & Murat, Y. S. (2019). A differential evolution algorithm-based traffic control model for signalized intersections. *Advances in Civil Engineering*, 2019, 1–16. <https://doi.org/10.1155/2019/7360939>
- [26] Baskan, O. (2019). A multiobjective bilevel programming model for environmentally friendly traffic signal timings. *Advances in Civil Engineering*, 2019, 1–13. <https://doi.org/10.1155/2019/1638618>
- [27] Civicioglu, P. (2013). Artificial cooperative search algorithm for numerical optimization problems. *Information Sciences*, 229, 58–76. <https://doi.org/10.1016/j.ins.2012.11.013>
- [28] Civicioglu, P. (2013). Backtracking search optimization algorithm for numerical optimization problems. *Applied Mathematics and Computation*, 219, 8121–8144. <https://doi.org/10.1016/j.amc.2013.02.017>
- [29] Civicioglu, P. (2012). Transforming geocentric cartesian coordinates to geodetic coordinates by using differential search algorithm. *Computers & Geosciences*, 46, 229–247. <https://doi.org/10.1016/j.cageo.2011.12.011>
- [30] Çivicioğlu, P., & Beşdok, E. (2019). Bernstein-search differential evolution algorithm for numerical function optimization. *Expert Systems with Applications*, 138.
- [31] Feliciani, C., Murakami, H., & Nishinari, K. (2018). A universal function for capacity of bidirectional pedestrian streams: Filling the gaps in the literature. *PLOS ONE*, 13, Article e0208496. <https://doi.org/10.1371/journal.pone.0208496>
- [32] Khan, K., Lobiyal, D. K., & Kılıçman, A. (2015). *Bezier curves and surfaces based on modified Bernstein polynomials* (arXiv:1511.06594). arXiv.
- [33] Azhari, F., Heidarpour, A., & Zhao, X.-L. (2018). On the use of Bernstein-Bézier functions for modelling the post-fire stress-strain relationship of ultra-high strength steel (Grade 1200). *Engineering Structures*, 175, 605–616. <https://doi.org/10.1016/j.engstruct.2018.08.088>
- [34] Mandal, B. N., & Bhattacharya, S. (2007). Numerical solution of some classes of integral equations using Bernstein polynomials. *Applied Mathematics and Computation*, 190, 1707–1716. <https://doi.org/10.1016/j.amc.2007.02.058>
- [35] Yousefi, S. A., Behroozifar, M., & Dehghan, M. (2011). The operational matrices of Bernstein polynomials for solving the parabolic equation subject to specification of the mass. *Journal of Computational and Applied Mathematics*, 235, 5272–5283. <https://doi.org/10.1016/j.cam.2011.05.038>