Beykoz Akademi Dergisi, 2025; 13(2), 487-516 ARAŞTIRMA MAKALESİ
Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

DYNAMIC BUS SCHEDULING IN ISTANBUL WITH HYBRID GASA[1]

Sara EL TANNİR[2], Yakup ÇELİKBİLEK[3]

Abstract

Transportation is a key element in any city, connecting the outermost suburbs to the center. Most studies done to elevate the transportation system are through optimizing the schedule of vehicles with respect to company's benefit. When the focus shifts to the riders, other projects gather ridership data to predict future demand. However, this project considers the rider's experience by optimizing a bus's schedule with the objective of minimizing passengers remaining at waiting stations using a hybrid genetic algorithm. The schedule will no longer be static, but dynamic, morphing periodically, to service every passenger possible while considering spaciotemporal factors, fleet size, vehicle capacity and more. Most scheduling problems use the genetic algorithm and its variations to solve scheduling problems, and this study integrates it with simulated annealing, since their combination can avoid poor local search of genetic algorithm and speeding its process. The proposed integrated genetic algorithm and simulated annealing has proven to be the best both genetic algorithm and simulated annealing in finding a solution that minimized the total number of remaining passengers in a day. This technique can be translated into other forms of transportation such as bus rapid transport, subways, trains and more.

Keywords: Dynamic Scheduling, Optimization, Genetic Algorithm, Simulated Annealing, Hybrid GASA
JEL Classification: C61, C63, R41

HİBRİT GASA İLE İSTANBUL'DA DİNAMİK OTOBÜS ÇİZELGELEME

Öz

Ulaşım, herhangi bir şehirde en dıştaki banliyöleri merkeze bağlayan önemli bir unsurdur. Ulaşım sistemini yükseltmek için yapılan çalışmaların çoğu, şirketin faydasına göre araçların programını optimize etmek yoluyla yapılır. Odak noktası yolculara kaydığında, diğer projeler gelecekteki talebi tahmin etmek için yolcu sayısı verilerini toplar. Ancak bu proje, hibrit genetik algoritma kullanarak bekleme istasyonlarında kalan yolcu sayısını en aza indirme amacıyla bir otobüsün programını optimize ederek yolcunun deneyimini dikkate alır. Çizelgeleme artık statik olmayacak, dinamik olacaktır ve uzaysal-zamansal faktörleri, filo büyüklüğünü, araç kapasitesini ve daha fazlasını göz önünde bulundurarak mümkün olan her yolcuya hizmet etmek için periyodik olarak değişecektir. Çoğu çizelgeleme problemi, çizelgeleme problemlerini çözmek için genetik algoritmayı ve onun varyasyonlarını kullanmaktadır ve bu çalışma genetik algoritma ile benzetilmiş tavlamayı entegre ederek, genetik algoritmanın zayıf yerel aramasından kaçınır ve süreci hızlandırır. Önerilmekte olan entegre genetik algoritma ve benzetilmiş tavlama, bir günde kalan toplam yolcu sayısını en aza indiren bir çözüm bulmada hem genetik algoritmadan hem de benzetilmiş tavlamadan daha iyi sonuçlar ürettiğini kanıtlamıştır. Bu teknik, hızlı otobüs taşımacılığı, metro, tren ve daha fazlası gibi diğer ulaşım biçimlerine uyarlanabilir.

Anahtar kelimeler: Dinamik Çizelgeleme, Optimizasyon, Genetik Algoritma, Benzetilmiş Tavlama, Hibrit GASA
JEL Sınıflaması: C61, C63, R41

[1] This study was produced from Sara El Tannir's masters thesis, conducted under the supervision of Assoc. Prof. Yakup Çelikbilek

[2] Sara El Tannir, Istanbul Aydin University, saratannir@aydin.edu.tr, ORCID: 0009-0008-2963-218X

[3] Assoc. Prof., Istanbul University, Faculty of Science, Department of Computer Science, yakup.celikbilek@istanbul.edu.tr, yakupcelikbilek@aydin.edu.tr, ORCID: 0000-0003-0585-1085

Beykoz Akademi Dergisi, 2025; 13(2), 487-516 ARAŞTIRMA MAKALESİ
Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

1. Introduction

Delving into dynamic scheduling must be accompanied by the discussion of static scheduling. Static scheduling follows the objective of forming a fixed schedule for each line (Ning et al., 2015). Predictably, dynamic scheduling is the opposite, and it falls under demand responsive transit where every bus line has a continuously changing schedule to suit their goals (Song et al., 2024). Such a process would be akin to having an entire service catered to the passengers' needs without sacrificing the profit reaped. These attributes, integral to the concept of smart cities, are no longer distant possibilities; their realization is now closer than ever.

The rise and widespread adoption of artificial intelligence (AI) across all aspects of life has become an undeniable reality, offering immense potential benefits. It is therefore natural for local governments to make use of those technologies for the services they provide. This is one of the many endeavors that are undertaken by many modern cities in their venturing into the new era of smart cities. AI and the Internet of things (IoT) are well known pillars of smart cities and can be effectively implemented in the domain of public.

Nowadays, many cities are reportedly struggling with providing an efficient public transportation system that can handle the ever-increasing population and their demands. The introduction of AI and IoT is a pivotal point in contending with them (Lu et al., 2020). Examples could be by applying AI in the following sub-sectors: predictive maintenance, route optimization, dynamic scheduling, passenger information systems, and fare optimization (Lu et al., 2020).

In the transportation sector, the main constituent of transportation is supply and demand, which goes both ways: the coordinators, e.g., the bus company, and the passengers interested in riding (ZhongXi et al., 2019). The supplier, i.e., the coordinator, aims to provide a good service that can keep up with the passengers' demands while also minimizing costs; their focus is on maintaining a suitable headway schedule. On the other hand, riders' main factors in perceiving the quality of service and customer satisfaction are waiting time, passenger density and slow speed (Ai et al., 2022).

Thus, the frequent problem for current operators is failing to maintain a punctual fleet schedule and size which simultaneously functions at full capacity (Ushakov et al., 2022). And this is where AI and IoT come in handy; that is, to bring balance between supply and demand (Ushakov et al., 2022). This can be done in multitude of ways, starting with placing sensors, card readers, and even cameras at stations and stops (Samasti, 2023; Welch and Widita, 2019).

This would be the ideal starting point enabling gathering of data, which can be used to monitor and later, analyze, the root causes of the prevalent issues in the sector.. The service of public transit is a pillar in modern cities, and integrating AI and IoT will mark a paradigm shift in that field (Vemuri et al., 2024).

According to TUIK (TUIK, 2024), the population of Istanbul has reached 15 million people and is set to maintain this trend in the upcoming years - a consequent of higher tourism, and in-migration rates. Such a high number surely predicts the critical role that public transportation is currently playing in netizens' everyday routine.

Along with the Istanbul municipality's (IBB) new law that requires it to expand the reach of the metropolitan's advantageous services to suburban areas (Arifoğulları and Alptekin, 2022). Istanbul's municipality is taking measures to provide for its residents by constructing an intricate network of transportation open to the public who hold an Istanbul card.

For the IBB, to gather all data describing the spatiotemporal movement of the buses, would have to cultivate an enormous amount of files worth of data. This leads to the need for research to study and utilize big data techniques to handle manipulating and filtering through the data.

There is a plethora of studies done on problem solving current public transportation challenges. However, a huge amount of intricate and clean data is necessary to tackle their usual objectives that range from service optimization, maintenance, health and safety, and passenger behavior, etc. (Welch and Widita, 2019). The methods of collecting data can be categorized as traditional and new. Where the traditional methods involves Automatic Fare Collection (AFC), Automatic Vehicle Location (AVL), and General Transit Feed Specification (GTFS) (Lu et al., 2020; Welch and Widita, 2019)

The fast advancement of technology has yielded new data collection approaches. These methods that are planned, used ethically, and prioritized maintaining anonymity, can revolutionize the way we gather data. Smart Phone Data contains the combination of GPS, Wi-Fi, and bus travel rate information. Data gathered via Bluetooth and Wi-fi technologies can peek into passenger behavior, their origin-destination (OD). In addition, biometric face recognition and social media data are also added value to data (Lu et al., 2020; Welch and Widita, 2019).

The data collection methods are used for both user-end and company-end optimization. One interesting topic is dynamic scheduling (Vemuri et al.,2024; Lu et al., 2020), a real- time reactive scheduling technique that responds to passenger demand during rush hours, traffic delays, incidents, and weather conditions. It is more agile and can keep up with the hectic urban lifestyle that modern cities face, unlike fixed schedules.

The purpose of this research is to provide an innovative and more efficient system to schedule public transportation following real time changes of entering and existing passengers. The goal is to devise an AI-based dynamic scheduling system using real time data monitoring, modelling, and provisioning the patterns of passengers' use of the system resulting in efficient control of its capacity and resources. Two heuristic algorithms are combined to create a hybrid algorithm that can better tackle such a feat: Genetic Algorithm (GA) and Simulated Annealing (SA), and dynamically optimize the fleet dispatching and trip monitoring.

2. Literature Review

Optimizing bus schedules based on demand is not a novel concept. However, traditional scheduling methods often fail to adapt to fluctuating passenger demand and external variables, leading to inefficiencies in public transportation systems. To address this, heuristic algorithms and mathematical programming methods have been widely employed to generate bus schedules with uniform departure intervals.

Optimization problems can be categorized as linear or non-linear. Most real-world problems are inherently non-linear due to the complexity of constraints, dynamic variables, and unpredictable fluctuations in demand. Traditional optimization techniques often struggle to provide feasible solutions within a reasonable timeframe. As a result, metaheuristic algorithms have been introduced, offering a more flexible and adaptive approach to solving these complex problems.

Before selecting an appropriate algorithm, it is crucial to analyze the nature of the problem. Optimization problems can be classified based on their characteristics: they may be single-objective or multi-objective, constrained or unconstrained, and either discrete or continuous. A careful assessment of these properties helps in determining the most suitable algorithm for a given problem.

Gradient-based methods, such as hill climbing, define the search direction based on the gradient of a function, selecting the nearest neighboring solution. These first-order algorithms are widely used in machine learning and neural networks due to their efficiency in optimizing smooth and differentiable functions. However, they tend to converge to the same solution if initialized from the same starting point, making them prone to getting stuck in local optima. This limitation reduces their effectiveness in solving complex, multi-modal optimization problems.

It is important to note that deterministic approaches do not introduce randomness. Once a starting point is chosen, these methods proceed by exploiting the nearest optimal solution without considering alternative regions of search space. This makes them efficient in well-structured environments but limits their adaptability to dynamic or uncertain conditions.

Population-based algorithms, on the other hand, employ multiple solutions simultaneously, enhancing the search process by maintaining diversity. This characteristic is evident in swarm intelligence algorithms, such as Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Artificial Bee Colony (ABC). These methods mimic the collective behavior of biological systems to explore a vast solution space and adapt to changing conditions, making them well-suited for dynamic optimization problems.

A key factor influencing an algorithm's performance is the balance between intensification (exploitation) and diversification (exploration). Exploitation focuses on refining solutions near local optima, improving convergence speed, whereas exploration encourages a broader search across the solution space, preventing premature convergence. Striking an optimal balance between these two strategies is essential for ensuring that the algorithm does not settle for suboptimal solutions. By integrating both strategies effectively, metaheuristic approaches can provide robust and adaptable solutions for complex scheduling and optimization tasks.

Among these, heuristic algorithms, particularly GA and its enhanced versions, are popular for optimizing bus schedules. For instance, Luhua et al. (Luhua, 2011) applied GA to optimize a function that considers both passenger costs and operator expenses, aiming to enhance service quality and reduce operational costs. Similarly, Qian et al. (2015) integrated a tabu search into GA, while Tang et al. (2020). proposed an improved NSGA-II to minimize passenger waiting times and operational costs. Researchers such as Wihartik et al. (2017), Li et al. (2018), and Tang et al. (2018) introduced enhanced GA approaches to further improve service quality. Gkiotsalitis and Alesiani (2019) developed an offline GA designed to optimize schedules while accounting for passenger demand and uncertain travel times.

An interesting paper published in 2015 by Ning et al. (Ning et al., 2015), applied a hybrid heuristic algorithm to solve a bus rapid transfer (BRT) problem. The aim was to enhance the local search ability of the evolutionary algorithm, while also integrating a fitness scaling function to increase the chances of the solutions with lower fitness to proceed to the next

Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

generation. The results of that integration showed a positive effect on the algorithm, in a sense where the optimal or near-optimal solution can be found at a faster rate and in a larger space. Nonetheless, they did not consider sudden passenger flow changes, especially since the schedule was not dynamic. Nevertheless, all the prior studies mentioned focus on creating an optimized static schedule. For dynamic scheduling, multiple studies done on that very idea in different parts of the world, most prominently, China. The majority, if not all, of the work focuses on minimizing waiting time of passengers. The waiting time of passengers is a pillar of customer satisfaction for bus companies (Mo et al., 2020). One example, a team of researchers in China (Mo et al., 2020) took on the task to optimize bus schedule frequency while minimizing waiting time of passengers, while also taking into consideration the constraints of limited vehicles. The problem was solved by introducing approximation algorithms and applying greedy algorithms.

Another study done in China (Lin and Tang, 2021), decided on adding an extra objective. Instead of only minimizing the passengers' waiting time for the vehicles, they maximized factors that interest the public transportation companies. They leveraged real-life data collected from card data and vehicle GPS and extrapolated the origin-destination data from real-time travel information. After, they applied enhanced quantum genetic algorithms to optimize scheduling vehicles and personnel data. They noticed improved overall efficiency, wider search space for the optimum solution, reduced memory space and an easier effort to generate an initial generation.

An alternative technique utilized in bus schedule optimization is through utilization is deep learning (Ai et al., 2022). More specifically, deep reinforced learning, to solve the problem that they formulated as a Markov-Decision-Problem with the purpose of scheduling the bus lines departure time. The decision to allow a bus to depart every minute depends on a reward system that measures a bus's full and empty load rate, remaining passengers at stations and their waiting time.

One of most recent studies regarding dynamic demand responsive transit scheduling is published by Song et al. in 2024 (Song et al., 2024). To lessen the complexity of their operations, the authors divided their research into two phases according to the resulting schedule type. The first objective used an evolutionary algorithm, minimizing vehicle cost and operational cost while mitigating CO2 emissions, resulting in a static schedule. Upon which, they further altered, with an insertion algorithm, in phase two to create a dynamic schedule with

Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

the same objectives and adding a penalty cost for violating time waiting time window and minimizing passenger rejection. To test their work, they applied it on demand with data gathered from Chaoyang District of Beijing. The stated results found that with dynamic scheduling, CO2 emissions were slightly lowered, more passengers were served but with slightly higher operation cost of buses.

A study that also tackled this subject recently is one proposing a millisecond controlled based approach for scheduling buses (F.Wang et al., 2024). Their work controller-based bus scheduling approach that makes decisions at each departure time by utilizing a duty type converter and a bus selector. The former dynamically converts the bus duty types to take advantage of the existing fleer available while the latter selects the ideal departure time. The controllers are optimized with a particle swarm optimizer (PSO). After multiple real-world applications, the authors concluded high quality schedules in with a decision time of 2 milliseconds.

Regarding studies done for Istanbul’s transit, the majority of transportation related focus on passenger prediction flow analysis (Utku and Kaya, 2022; Sevim et al., 2022; Utku and Kaya, 2023). Both papers by Utku and Kaya used neural network techniques to estimate future number pf passengers. While Sevim et al. (2022) took advantage of real data provided by the IETT, the same one was used in this study and incorporated as historical data in a machine learning model to predict future demand.

There is a clear lack of leading academic work that addresses the issue of advanced scheduling using data gathered by smart cards, payments and sensors in Türkiye in general and Istanbul specifically. Hence, this study will spearhead this movement.

Table 1. Literature Review Summary

Author(s)	Year	Objective	Methodology	Scheduling Type
Luhua et al.	2011	Optimize bus schedule considering passenger cost and operator payment.	GA	Static
Qian et al.	2015	Reduce operational costs and passengers' waiting time.	GA with embedded Tabu Search	Static
Yang et al.	2020	Reduce operational costs and passengers' waiting time.	Improved NSGA-II	Static
Wihartiko et al.	2017	Improve transit service quality	Improved GA	Static
Li et al.	2018	Improve transit service quality	Improved GA	Static
Tang et al.	2018	Improve transit service quality	Improved GA	Static

Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

Gkiotsalitis and Alesiani	2019	Optimize bus schedules considering passenger demand and uncertain travel time.	Improved GA	Static
Ning et al.	2015	Minimize passenger waiting time.	Hybrid GASA and fitness scaling	Dynamic
Mo et al	2020	Minimizing passenger waiting time.	Greedy Algorithm	Dynamic
F. Wang et al.	2024	Optimizing bus departure schedule	Particle Swarm Optimizer	Dynamic

3. Methodology

As previously mentioned, the approach in this study leverages artificial intelligence to dynamically minimize the number of passengers left waiting at bus stations. This optimization process is triggered whenever there is a significant fluctuation in the number of arriving or departing passengers, ensuring that the bus schedule adapts in real time to changing demand.

For the optimization process, heuristic algorithms are use. These algorithms can be grouped into four (Jebari and Madiafi, 2013):

1. Evolutionary algorithms simulate the tendencies of living organisms in the animal kingdom, where they rely on the foundational concepts of Darwinian-like natural selection. The most famous is genetic algorithms. Others include differential evolution.
2. Physical-based algorithms; conceptualized on the fundamental principles of Newton's law of physics. Gravitational Search Algorithm (GSA), imitating the gravitational interactions between masses.
3. Swarm-based Algorithms; by emulating the swarms of insects/animals, the algorithm starts off differently. So, instead of taking one initial solution and generating a neighboring solution from that, multiple particles are set and work as a self-organized and decentralized way to reach the optimum solution.
4. Human-based algorithms are based on mathematical models and intelligently devised procedures mimicking the characteristics of human activities.

In the following sub-sections, a detailed explanation of two key optimization algorithms are provided. Genetic Algorithm (GA) is inspired by the principles of natural selection and evolution, using crossover, mutation, and selection mechanisms to iteratively improve solutions. Simulated Annealing (SA), on the other hand, is a probabilistic technique that mimics the annealing process in metallurgy to escape local optima and explore a broader solution space.

Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

Furthermore, a hybrid approach known as GASA (Genetic Algorithm-Simulated Annealing) combining the strengths of both methods is introduced. By integrating GA's ability to explore diverse solutions with SA's capacity to fine-tune and avoid premature convergence, the GASA approach aims to achieve a more effective and adaptive scheduling optimization.

3.1. Genetic Algorithm, Simulated Annealing, Hybrid GASA

3.1.1. Genetic algorithm

GA is a well-known algorithm that was inspired by the process of evolution - proposed by J.H. Holland in 1992 (Holland, 1992). Where chromosomes are the representation of solutions in space. The solution goes through other biological inspired operations from the process of evolution: An objective function is used to assign value for all the chromosomes in the population (Michalewicz and Schoenauer, 1996). Then selection of the chromosomes is done based on fitness, where the solutions with the highest, or lowest if it is a minimizing case, values are chosen for the next operation. Crossover, where a random spot in the chromosome set is chosen and is changed the subsequences between chromosomes to create offsprings. In mutation, some bits of the chromosomes will be randomly flipped based on probability (Michalewicz and Schoenauer, 1996).

The chromosome representation, selection, crossover, mutation, and fitness function computation are the key elements of GA. The procedure of classical GA is as follows:

1. Set the maximum number of iterations (generations) allowed, population, as well as other variables and constancies.
2. Initial population is generated to represent the first generation, where each individual, also known as chromosome or solution, are evaluated with the evaluation function.
3. The chromosomes are selected according to their scores.
4. The selected solutions are crossed over with each other.
5. Mutation is applied to a certain extent to preserve population diversity.
6. New Generation is produced and replaces the old generation.
7. The process repeats till a maximum number of iterations is reached.

There are several ways to diversify GA, and that is by changing the biological operators. Selection, phase translates reproduction operator in real life (Fox and McMahon, 1991; Freisleben and Merz, 1996) and puts pressure on the criteria for selection which in turn affects the convergence rate of GA. The selection technique includes Roulette Wheel, Rank selection, tournament selection, Boltzmann selection, Stochastic Universal Sampling, and Elitism (Katoch et al., 2021).

The roulette wheel places the solutions onto an imaginary spinning wheel, where each solution's section of the circle correlates with its fitness, the fittest solution has the biggest probability of being selected, and so on. Other selection methods modify the former operation to decrease errors caused by stochasticity. Tournament selection is one that takes inspiration from the roulette selection. Where the solutions are distributed the same, but instead of choosing one at a time, a couple are chosen, and the solution with the highest fitness value gets to contribute to the next generation (Jebari and Madiafi, 2013). Stochastic universal sampling (SUS) is another method that relies on the concept of roulette wheel, but it offers equal chances for the solutions to be selected for crossover (Abdulal and Ramachandram, 2011). Moreover, selection by rank was introduced. The chromosomes are ranked according to their fitness scores (Abdulal and Ramachandram, 2011). Boltzmann Selection is based on the Monte Carlo simulation, where its entropy and sampling techniques are aided in solving the premature convergence (Lee, 2003). However, since its running time is faster, information loss of the best fit solution can occur. This disadvantage can be managed by Elitism (Saini, 2017), proposed by De Jong (1975). This method guarantees the elites, those with best evaluation score, move forward to participate in the next generation.

Crossover operation procedures mostly differ on the point(s) of segmenting the chromosome or genetic makeup for exchanging data with the other parent (Soon et al.,2013). For example, single, K-point crossover. Contrary to the former methods, Uniform Crossover randomly picks locus and switches them with the other parent's respective gene at that locus. More operations exist; however, the child succeeds in their alleles by inheriting alternatively between its antecedents.

Mutation is a crucial operator in genetic algorithms, playing a vital role in maintaining genetic diversity and preventing population stagnation due to inbreeding. It introduces variations into the population, ensuring that the search process does not become trapped in local optima. The mutation process consists of two key elements: the probability of mutation and the nature of the

Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

outlier produced after the mutation operation. The probability of mutation determines how frequently genetic modifications occur, directly influencing the algorithm's ability to explore new solutions. Meanwhile, the characteristics of the mutated individuals affect the overall diversity of the population and the likelihood of discovering an optimal solution. By modifying encoded genes and evaluating entire generations simultaneously, mutation enhances the algorithm's global search capability. This ability to introduce new genetic variations ensures a more comprehensive exploration of the solution space, increasing the chances of converging toward the most optimal solution.

Nonetheless, in each generation, the most optimal solutions are typically selected to undergo crossover, combining their genetic information to produce the next generation of solutions. However, this process can sometimes lead to the unintended loss of desirable solutions, as less fit yet potentially valuable genetic variations may be eliminated too early in the evolutionary process.

3.1.2. Simulated Annealing

First introduced in the 1980's, SA was pioneered by Kirkpatrick et al. (Kirkpatrick et al., 1983) and inspired by the annealing process of metals. It was conceptualized on the foundation of blacksmiths' work on metals. That is, annealing is a heat treatment undergone onto metals to make the rigid matter more malleable and ductile for manipulation and handling. The process reduces the risk of forming cracks, or even breaking, the metal when strong forces are applied (Tech Steel & Materials, 2025). The process starts with the metal to temperature to above the material's recrystallization point, then letting it cool gradually and recrystallize back. This method abets rearranging the metal's grains into a series with no defects. (TWI, 2025).

So simulated annealing is mimicking the technique and applying it in an optimization problem. With the temperature acting as an entropy, hence, as the temperature decreases, so does the randomness. This enhances the algorithm's search technique and allows it to showcase a sort of balance between exploration and exploitation (Baeldung, 2025).

The first step includes setting the constants, such as initial temperature, cooling rate (denoted with the symbol alpha), and the number of iterations. With each pass of the algorithm, the temperature is decreased by the cooling rate.

$$T_{k+1} = \alpha \cdot T_k \tag{1}$$

Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

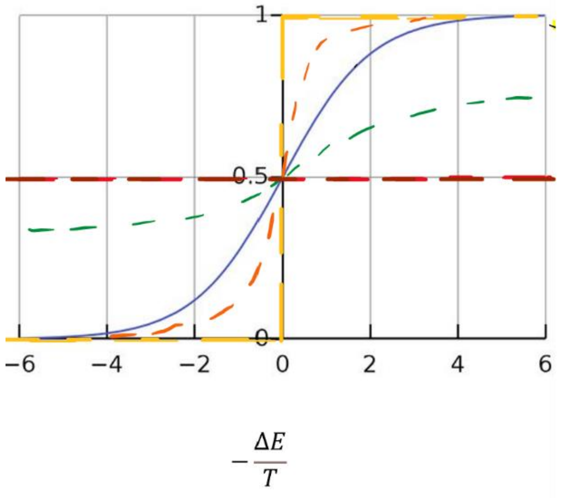
The second step involves generating a random initial solution, which serves as the starting point for the optimization process. From this point, a neighboring solution is derived by making slight modifications to the current solution. However, the acceptance of this new solution depends on an evaluation process, which assesses its quality based on predefined criteria.

$$\Delta E = E(x_{k+1}) - E(x_k) \tag{2}$$

, where k is the current/initial solution and k+1 is the next/neighbor solution. If this evaluation illustrates improvement in solution optimization, i.e. $\Delta E > 0$. Then the neighboring solution is accepted as the new solution. On the other hand, if the difference in energy is negative, then the value is then inserted, along with temperature, into a sigmoid function, which returns a probability.

$$P(\Delta E) = \frac{1}{e^{\Delta E/T}} \tag{3}$$

It is imperative to note that the sigmoid function's output changes extremely on the current temperature. For example, in the preliminary stages of the problem, T is still relatively high, the sigmoid function's slope decreases, garnering a relatively high probability- this is particularly evident when compared to solutions with higher evaluations. This causes the algorithm to see both solutions as acceptable. This allows the algorithm to behave like Random Walk. Conversely, when the temperatures are cooler, the sigmoid function tends to look like a stepwise function, enforcing harsher acceptances. The acceptable solution will be the one with the higher probability- only exploitation, like Hill Climbing.



Graph 1. Sigmoid function fluctuation against temperature

Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

3.1.3. GASA hybrid algorithm

While GA is known to handle scheduling problems very well and is able to have a widespread/global search area, one of its main drawbacks, prematurity cannot be brushed aside (Ning et al., 2015; Katoch et al., 2021). In addition, it is common for the algorithm to fall into local optima points in the solution space. On the other hand, SA has narrow reach in the search space, but during the preliminary stages it accepts solutions with lower fitness values due to the high entropy. Combining those GA and SA, to produce a hybrid metaheuristic algorithm that theoretically can bypass both their individual shortcomings.

Integrating those algorithms together can exist in 3 different ways (Welch and Widita, 2019):

1. SA is integrated into GA at the selection stage, where the selection is operated according to the principle of combining simulated annealing and simulated binary.
2. SA is integrated in the initial and final stages of the population; GA is used to evolve the population and then the SA is used to further adjust the group solution. SA comes after the crossover and mutation functions are complete.
3. SA is integrated after the GA step. They operate alternatively, where GA will do an initial global search, the solution obtained will be the simulated annealing initial population, then via SA local search is carried out. obtained solution from SA will be the initial solution for GA.

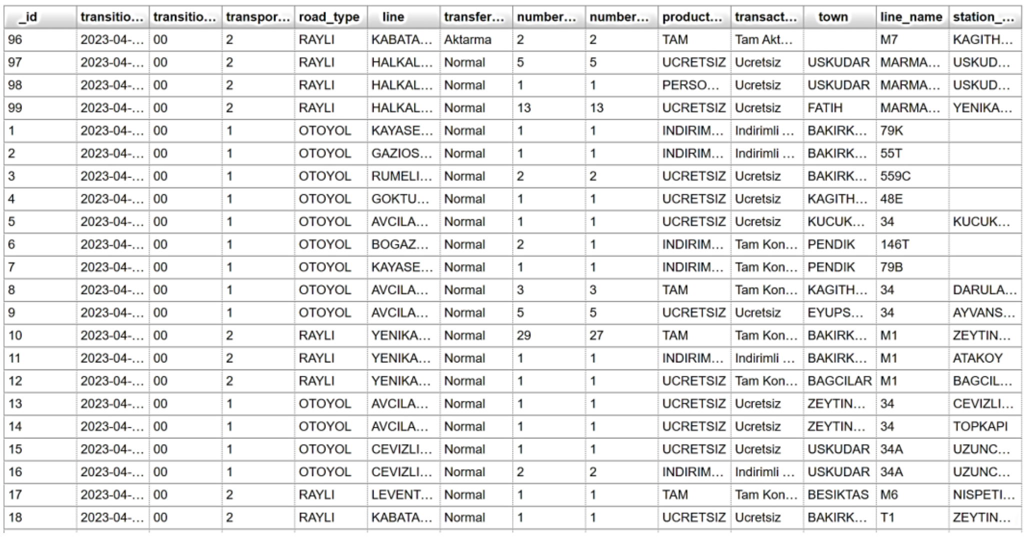
The hybridization of those algorithms to create GASA, will result in a real-time scheduling tool for problems with time and resource constraints (Wei Pu and Yi-Ren Zou, 2002).

3.2. Data Preparation

The data source is downloaded from Istanbul's Municipality open-source database (https://data.ibb.gov.tr/en/dataset/hourly-public-transport-data-set). The raw data is illustrated in the image below.

Beykoz Akademi Dergisi, 2025; 13(2), 487-516 ARAŞTIRMA MAKALESİ
Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

Figure 1. Raw data



_id	transitio…	transitio…	transpor…	road_type	line	transfer…	number…	number…	product…	transact…	town	line_name	station_…
96	2023-04-…	00	2	RAYLI	KABATA…	Aktarma	2	2	TAM	Tam Akt…		M7	KAGITH…
97	2023-04-…	00	2	RAYLI	HALKAL…	Normal	5	5	UCRETSIZ	Ucretsiz	USKUDAR	MARMA…	USKUD…
98	2023-04-…	00	2	RAYLI	HALKAL…	Normal	1	1	PERSO…	Ucretsiz	USKUDAR	MARMA…	USKUD…
99	2023-04-…	00	2	RAYLI	HALKAL…	Normal	13	13	UCRETSIZ	Ucretsiz	FATIH	MARMA…	YENIKA…
1	2023-04-…	00	1	OTOYOL	KAYASE…	Normal	1	1	INDIRIM…	Indirimli …	BAKIRK…	79K	
2	2023-04-…	00	1	OTOYOL	GAZIOS…	Normal	1	1	INDIRIM…	Indirimli …	BAKIRK…	55T	
3	2023-04-…	00	1	OTOYOL	RUMELI…	Normal	2	2	UCRETSIZ	Ucretsiz	BAKIRK…	559C	
4	2023-04-…	00	1	OTOYOL	GOKTU…	Normal	1	1	UCRETSIZ	Ucretsiz	KAGITH…	48E	
5	2023-04-…	00	1	OTOYOL	AVCILA…	Normal	1	1	UCRETSIZ	Ucretsiz	KUCUK…	34	KUCUK…
6	2023-04-…	00	1	OTOYOL	BOGAZ…	Normal	2	1	INDIRIM…	Tam Kon…	PENDIK	146T	
7	2023-04-…	00	1	OTOYOL	KAYASE…	Normal	1	1	INDIRIM…	Tam Kon…	PENDIK	79B	
8	2023-04-…	00	1	OTOYOL	AVCILA…	Normal	3	3	TAM	Tam Kon…	KAGITH…	34	DARULA…
9	2023-04-…	00	1	OTOYOL	AVCILA…	Normal	5	5	UCRETSIZ	Ucretsiz	EYUPS…	34	AYVANS…
10	2023-04-…	00	2	RAYLI	YENIKA…	Normal	29	27	TAM	Tam Kon…	BAKIRK…	M1	ZEYTIN…
11	2023-04-…	00	2	RAYLI	YENIKA…	Normal	1	1	INDIRIM…	Indirimli …	BAKIRK…	M1	ATAKOY
12	2023-04-…	00	2	RAYLI	YENIKA…	Normal	1	1	UCRETSIZ	Tam Kon…	BAGCILAR	M1	BAGCIL…
13	2023-04-…	00	1	OTOYOL	AVCILA…	Normal	1	1	UCRETSIZ	Ucretsiz	ZEYTIN…	34	CEVIZLI…
14	2023-04-…	00	1	OTOYOL	AVCILA…	Normal	1	1	UCRETSIZ	Ucretsiz	ZEYTIN…	34	TOPKAPI
15	2023-04-…	00	1	OTOYOL	CEVIZLI…	Normal	1	1	UCRETSIZ	Ucretsiz	USKUDAR	34A	UZUNC…
16	2023-04-…	00	1	OTOYOL	CEVIZLI…	Normal	2	2	INDIRIM…	Indirimli …	USKUDAR	34A	UZUNC…
17	2023-04-…	00	2	RAYLI	LEVENT…	Normal	1	1	TAM	Tam Kon…	BESIKTAS	M6	NISPETI…
18	2023-04-…	00	2	RAYLI	KABATA…	Normal	1	1	UCRETSIZ	Ucretsiz	BAKIRK…	T1	ZEYTIN…

Source: https://data.ibb.gov.tr/en/dataset/hourly-public-transport-data-set

Columns include:

transition_date: denotes the date at which a transaction is made via the Istanbul Card.

transition_hour: denotes the hour at which a transaction is made via the Istanbul's Card.

transition_type_id: classifies the type of transport taken. Classed into 3 categories.

road_type: describes each category; 1 = OTOYOL, 2 = RAIL, 3 = SEA.

line: name of the line of the transport taken.

tranfer_type: determined whether the passenger's card that is used has.

number_of_passage: number of trips a line runs per hour.

number_of_passenger: number of passengers who rode the transport.

The data files are in comma separated values type file, and are separated by month, each file represents a month of the year.

Apache Hadoop is an open-source software tool. It grants data scientists the ability to tame big data, both structures and semi-structures, by leveraging a network of computers to work in batches. It is proven to scalable and cost effective, relies on disk storage, works in batches.

Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

Apache Spark, on the other hand, is an open-source data processing engine for big data. It is similar to Hadoop yet relies on high quantities of RAM and presents results at faster rates. It utilizes real-time processing and a live unstructured data stream.

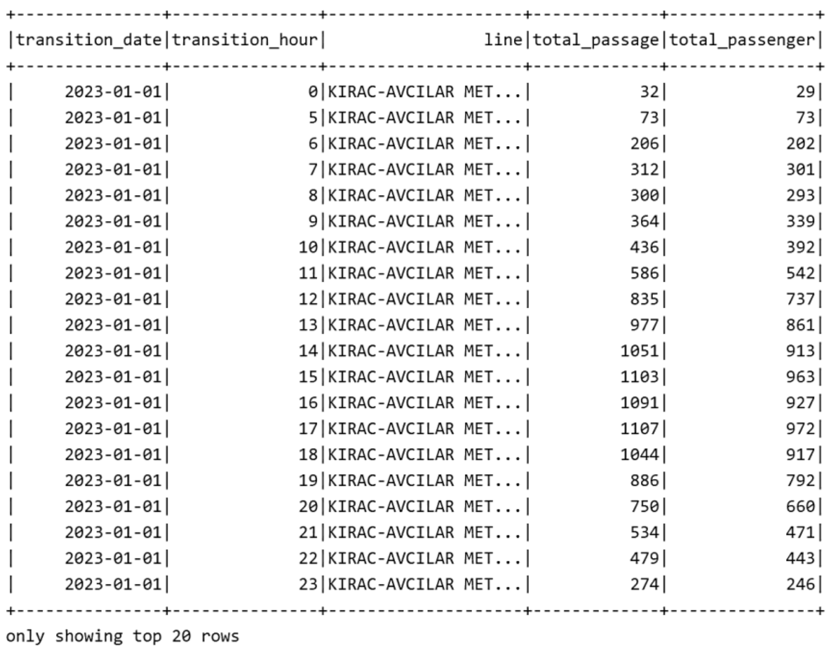
The Databrick environment is chosen because it is built on top of Apache spark and is optimized for various performances. It is interactive with data science tools and can run multiple versions of Spark, and is optimized for cloud storage access (azure, AWS), since it has a built-in file system. All those factors add up to make Databricks an ideal beginner friendly environment for big data workspace (Databricks, 2024).

Preprocessing and data extracting steps.

1. Combine csv files by year.
2. Remove null values, if any are found.
3. Change the column’s types into their correct datatypes.
4. Make sure the data is ordered chronologically according to date then hour.
5. Extract data relative to the bus line of interest.
6. Aggregate the data by summing the number of passengers of instances with the same date, hour and line.
7. Save.

The next step includes isolating the lines by name. The figure below shows the clean dataset of line 429A, better known as Kirac-Avcilar Metrobus.

Figure 2. Processed and filtered data for bus line 429A



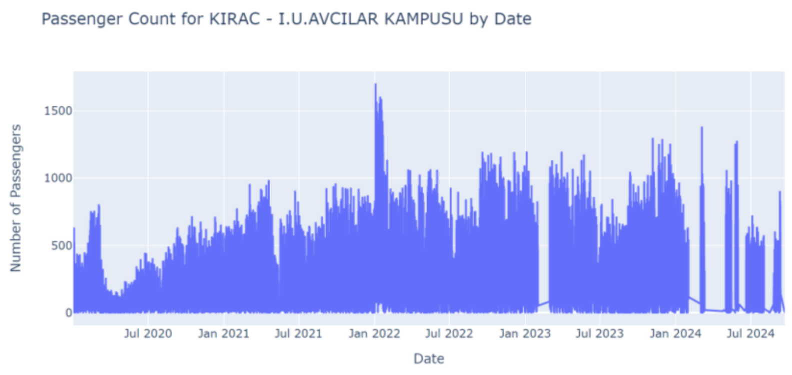
```
+---------------+---------------+--------------------+-------------+---------------+
|transition_date|transition_hour|                line|total_passage|total_passenger|
+---------------+---------------+--------------------+-------------+---------------+
|     2023-01-01|              0|KIRAC-AVCILAR MET...|           32|             29|
|     2023-01-01|              5|KIRAC-AVCILAR MET...|           73|             73|
|     2023-01-01|              6|KIRAC-AVCILAR MET...|          206|            202|
|     2023-01-01|              7|KIRAC-AVCILAR MET...|          312|            301|
|     2023-01-01|              8|KIRAC-AVCILAR MET...|          300|            293|
|     2023-01-01|              9|KIRAC-AVCILAR MET...|          364|            339|
|     2023-01-01|             10|KIRAC-AVCILAR MET...|          436|            392|
|     2023-01-01|             11|KIRAC-AVCILAR MET...|          586|            542|
|     2023-01-01|             12|KIRAC-AVCILAR MET...|          835|            737|
|     2023-01-01|             13|KIRAC-AVCILAR MET...|          977|            861|
|     2023-01-01|             14|KIRAC-AVCILAR MET...|         1051|            913|
|     2023-01-01|             15|KIRAC-AVCILAR MET...|         1103|            963|
|     2023-01-01|             16|KIRAC-AVCILAR MET...|         1091|            927|
|     2023-01-01|             17|KIRAC-AVCILAR MET...|         1107|            972|
|     2023-01-01|             18|KIRAC-AVCILAR MET...|         1044|            917|
|     2023-01-01|             19|KIRAC-AVCILAR MET...|          886|            792|
|     2023-01-01|             20|KIRAC-AVCILAR MET...|          750|            660|
|     2023-01-01|             21|KIRAC-AVCILAR MET...|          534|            471|
|     2023-01-01|             22|KIRAC-AVCILAR MET...|          479|            443|
|     2023-01-01|             23|KIRAC-AVCILAR MET...|          274|            246|
+---------------+---------------+--------------------+-------------+---------------+
only showing top 20 rows
```

Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

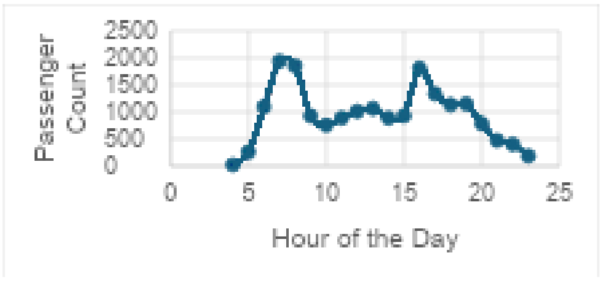
3.3. Data Exploration

In this sub-section, the data for preliminary investigation is explored.

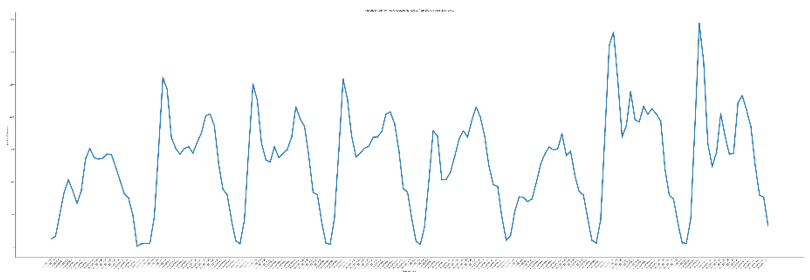
Graph 2. All time hourly passenger demand for line 429A



Graph 3. Hourly passenger demand in a day



Graph 4. Hourly passenger demand for line 429A in a week



At a first glance of Graph 2, the passenger count for the 429A bus from the years 2020 to 2024, the effects of the covid-19 pandemic is evident in the first half of 2020. The sudden drop in passengers is due to the forced quarantine that limits peoples’ activity in public. After a few months, a slow rise of passengers can be witnessed till January of 2022, when a sudden unexplained spike of passengers may be attributed to the municipality’s data collection technique was modified, since that exact spike is also evident in other bus lines. Towards the right side of the graph, there exists several discontinuities, that immediately should be disregarded for choosing.

Hence, choosing a random day to pick out the passenger trend should be from the year 2023, to avoid Covid-19 transportation restrictions' effects, and to steer from the incomplete data of 2024. The day that was chosen was in November 2023, and its passenger count can be seen in Graph 3. The passenger count per hour of the day is illustrated and shows the two periods of spikes, those pertaining to the morning and the evening rush hours. This is attributed to the high influx of residents taking public transportation to start their day for business or education, and the end of the day when business hours end.

To verify the trend in Graph 3 is recurring, Graph 4 is plotted to observe the passenger count of the bus line in a period of a week. Each day is indeed characterized by its dual peaks, except Saturday and Sunday, which should be studied separately due to the bus's different working hours as well as the lack of demand.

The bus line chosen to study is line 429A, Kirac-Istanbul University campus. The line is characterized by 51 stations, and, according to Istanbul Electricity Tram and Tunnel establishment (IETT) (IETT, 2024), needs around 55 minutes to continue one way path. There are a couple alternative routes to the original route, however, those are ignored.

The option for that line was by design; it is one of the shorter tracks with a smaller number of stations. The route does not pass through the innermost part of the city, where the number of passengers can be heavily affected by the tourism season, but generally on the suburban range with only a few attractions along that can weigh the passengers' behavior. The route of this line also can be sectioned into two, the former passing along the E-5 highway, while the latter is where the track diverts into Esenyurt neighborhood.

The bus generally takes 50 minutes to complete a one-way trip. However, the website mentioned that the time usually alters by 10 to 20 minutes to account for rush hours.

For applying the extracted dataset to an algorithm additional data collection is required. The dataset obtained from the IETT's dataset website only provides the number of passengers at a certain time. More data is needed to stipulate the spatiotemporal data of the bus line. Hence, the static schedule of the line is extracted from IETT. The schedule provided is divided into 3 parts.

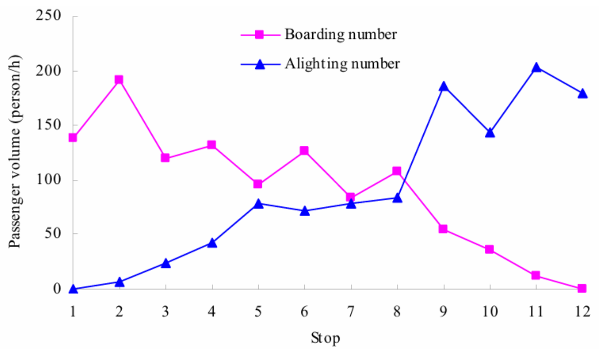
Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

It seems each part pertains to the timetable for business days, Saturday and Sundays. Note that the weekends are separate.

After tracking live data from Moovit website (Moovit, 2024), data for time taken to travel from one station to the next, during the day. The average time can be inferred to be around 1 to 2 minutes. To facilitate the running time of the algorithm, and avoid having many iterations for every minute, the stations are grouped according to the time interval chosen, 5 minutes. Hence, by grouping the stations by the previously allotted time interval, the stations will be clustered into 17 stations. The 5-minute interval was chosen, because a wider time slot will result in oversimplicity due to the small number of stations.

Unfortunately, the data is limited without any precision, as in, lack of data of the number of riders arriving and exiting the bus station. Therefore, the data for each time interval within each hour is distributed to represent positive and negative entries.

Graph 5. BRT station by station passenger demand



Source: Chuanjiao et al.,2008.

According to the study of Chuanjiao et al. (2008), there is a behavioral pattern of riders in public transportation, in their case, BRT. At the first few stations, the frequency of boarding passengers is at the highest. Conversely, the number of alighting passengers is zero, and barely higher. As the trip continues along its route, there is a noticeable decrease in the number of disembarking passengers, to the point where, down the middle of the route, the numbers of alighting and disembarking are balanced and relatively within the same range of 70 to 115 passengers per station. After a few stops, the roles are reversed, where the disembarking passengers increase – behaving inversely to boarding passengers whose values decrease to zero in the last few stops.

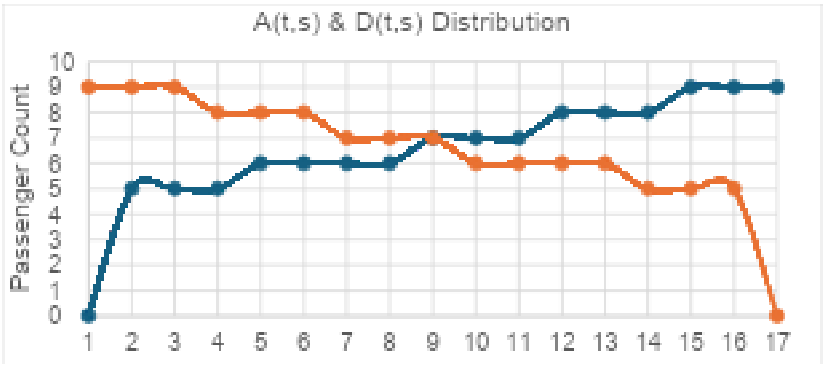
Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

This provides insight into passengers' distribution among stations in public transit. Hence, the hourly passengers' demand is distributed in a similar manner among the 17 stations as is illustrated in Graph 6.

An important note to bear in mind is that, logically, the number of riders embarking at the last stop cannot be positive, unless the second trip, going in the opposite direction has started. Similarly, there can be no disembarking passengers at the zeroth station, since the bus would be originally empty.

It is important to mention, for a more accurate representation of riders' behavior, clear data is needed, or at least, previous study on the trends for the bus line. Variation in the distribution can have effects on the schedule.

Graph 6. Passenger demand distribution by grouped station



3.4. The Proposed Model and Pseudo Code

The model to be solved consists of an objective function along with a set of constraints that must be satisfied. Once these are defined, we present the pseudo-code for the algorithm, which outlines the step-by-step procedure required to achieve the desired objective.

3.4.1. The Proposed Model

Objective functions: Minimization of total remaining passengers in a day.

Condition 1: [C1] – No exceeding the fleet size. Every 2 hours, the approximate period the bus takes to continue a trip from and to the depot should utilize less than or similar number of the buses available at the depot.

Condition 2: [C2] – Maximum capacity of the bus. The bus can fit a certain number of passengers at a given time. Once the limit is reached, the excess passengers are left stranded.

Assumptions:

1. Bus type, hence, capacity is uniform. So, all buses in the fleet can host the same number of passengers.
2. Delays caused by traffic are average. Further details in the discussion section.
3. FIFO rule for passengers.
4. The algorithm will simulate the bus road as if it's a one-way bus. However, to account for the other route and avoid exhausting the fleet, a constriction is added where the number of buses used every 2 hours does not pass the fleet size.
5. Passenger distribution according to time and space is set by the author.
6. The bus operates at constant speed.
7. Time of passenger alighting and boarding are negligible.

3.4.2. The Flow Chart

t: time interval [0,215], representing 5 minutes within the servicing hours.

s: station [0,16]

C: capacity constant

F: fleet size

A(t,s): rider arriving at time t, and station s.

E(t,s): embarking riders at t,s.

D(t,s): disembarking rider at t,s.

B(t,s): available spots in the bus that arrives at t,s.

R(t,s): remaining riders at t,s.

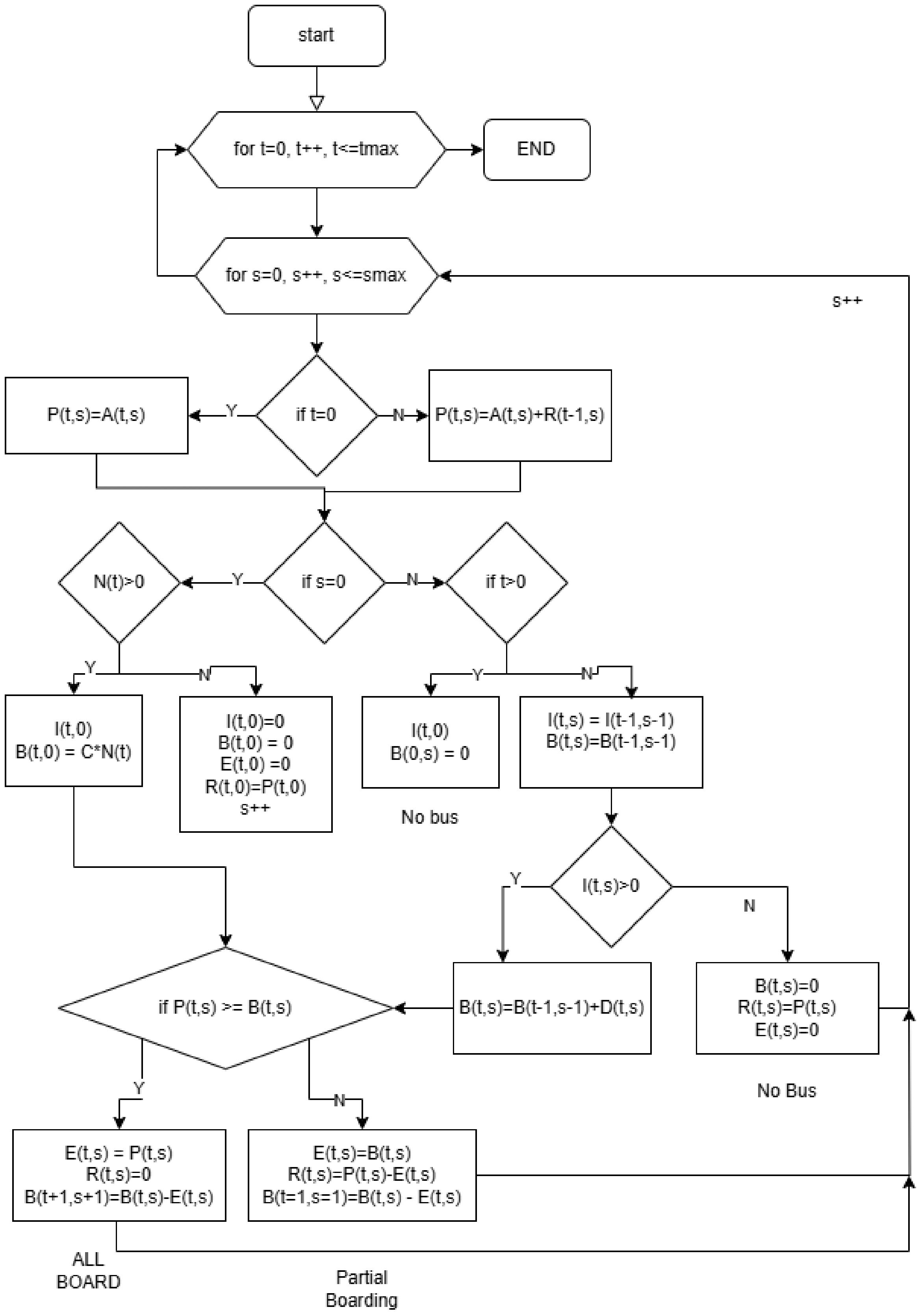
R(t): the sum of remaining passengers at each time slot or column.

The flowchart of the algorithm by using these notations is given below.

See appendix A for pseudo code.

Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

Figure 3. Evaluation function flow chart

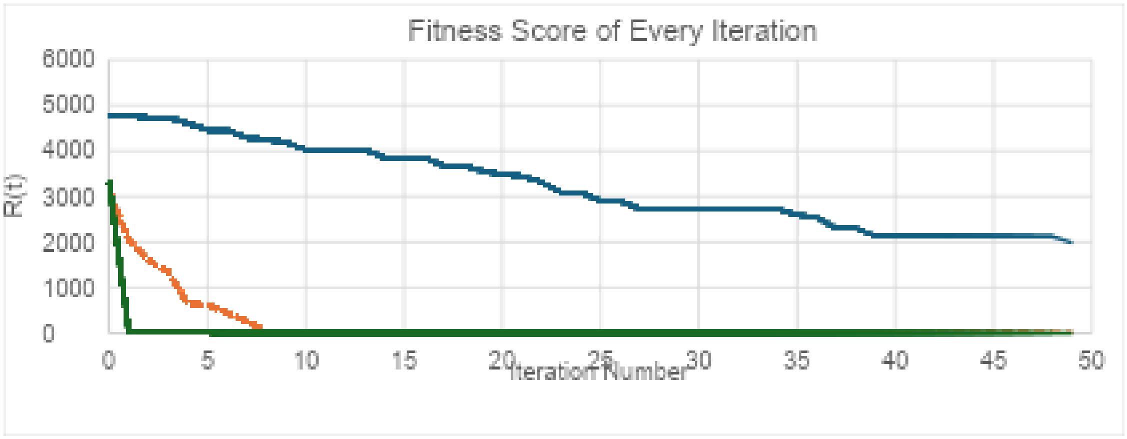


4. Findings and Discussions

Runing the three algorithms provided different places looking for better dispatching schedule solutions that attempt to minimize the total number of passengers who ended up remaining in any stations who didn't find places to embark on the bus.

The graph below illustrates the trend lines of the summation of the remaining passenger, R, for the best fit solution of each generation for each algorithm. Blue represents SA, orange, GA and their hybridization in green. The objective function is to minimize R.

Graph 7. Comparison of algorithms' efficiency



The initial points of trend lines purely depend on the best random solution in the first randomly generated population. Or, in the case of SA, the initial solution evaluation.

However, the rate of decline of R is crucial to witness the effectiveness of the methods in closing the gap of the optimal solution. The rate of decline for both the GA and its Hybrid decreases rapidly, with the latter exhibiting a sharper, nearly linear trend. On the other hand, SA's decline is steady and continuous throughout the entire run, suggesting a slower convergence and less effective optimization.

GA and Hybrid GASA algorithms significantly outperform SA in terms of convergence speed and solution quality. The Hybrid GASA performs slightly better than GA, potentially leveraging the strengths of both methods for enhanced optimization. Meanwhile, SA is slower and less effective in reducing the fitness score, indicating a less efficient approach for this problem.

The runtime duration for each algorithm varies significantly from one another. Starting with SA, as expected, it took the least amount of time to complete 50 iterations- approximately 10

Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

to 15 minutes. However, classical GA took roughly 40 minutes to reach the near-optimal solution and performed drastically better than its forerunner. Finally, the hybrid algorithm's duration took up to 1.5 hours, where $R(t) = 0$.

Table 2. $R(t)$ values for each algorithm in every 5 iterations

Iteration	GA	SA	HYBRID
0	3125	4778	3312
4	648	4613	4
9	23	4191	0
14	17	3858	0
19	17	3681	0
24	13	3084	0
29	13	2721	0
34	13	2721	0
39	13	2721	0
44	13	2115	0
49	9	1984	0

4.1. Advantages

- An additional objective—minimizing empty spaces in the vehicle—can be seamlessly integrated into the model. This ensures that the optimization process considers both passenger demand and the company's supply capacity, including fleet size, leading to a more efficient transportation system.
- The model is highly adaptable, allowing for modifications in fleet size and vehicle capacity. This flexibility makes it possible to account for factors such as temporary vehicle shortages due to maintenance or variations in bus types, ensuring optimal resource allocation.
- Because the model accommodates different bus types, it can be extended to dynamic scheduling across various modes of transportation. This adaptability makes it applicable not only to buses but also to other public transit systems, such as trains and subways.

4.2. Limitations

- The algorithm is currently oversimplified, as it only considers the number of remaining passengers. A more realistic approach would be to prioritize passenger waiting time, which is one of the most critical factors in customer satisfaction. Since waiting time is directly influenced by trip delays caused by traffic conditions, optimizing for it would provide a more accurate measure of service quality. Reducing passenger waiting times could significantly enhance customer experience, ultimately attracting more passengers to the transportation service.

- To ensure accuracy, implementing the algorithm on a real bus line must be supported by a comprehensive analysis of passenger demand. If station-by-station passenger data is unavailable, identifying demand trends becomes essential. In this study, the station-by-station distribution was primarily based on assumptions and general observations, which may introduce limitations in real-world applicability. A more data-driven approach would enhance the model's reliability and effectiveness.

4.3. Future Work

Future work required before real-life implementation in modern cities includes adding an additional objective to enhance optimization. Additionally, collecting more accurate and comprehensive data on passenger behavior is crucial to improving demand prediction based on daily and hourly fluctuations. Key variables to consider include weather conditions, major events, and even outbreaks of highly contagious diseases, all of which can significantly impact passenger flow and travel patterns.

Considering traffic delay data can be leveraged from Moovit (Moovit, 2024), by live tracking the average total trip time it takes for the bus to finish a one-way trip change during the day. This can be attributed to the traffic caused by privately owned mobiles, buses, and miscellaneous vehicles that pass the same route as the bus in question. Moreover, the path of the line passes through the e-5 highway. Hence, the working hours of the bus will be divided into sections according to the duration it takes. The first section requires 65 minutes, and its working hours ranges are 5-6 and 20-23. The second section requires 80 minutes, this is attributed to the rush hours which are also divided into two parts of the day; the morning rush hours 6-9 and the evening 14-20. The third section requires 65 minutes to complete a trip, and it encompasses the working hours, excluding the rush hours; ranges include the hours of 10 to 14.

Although this feature was not incorporated into the current algorithm, it can be integrated by utilizing GPS tracking in vehicles combined with live continuous data streaming of passenger entry and exit patterns. This real-time data would enhance the system's ability to dynamically adjust schedules based on actual ridership, improving overall efficiency and responsiveness.

Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

Furthermore, with the aid of smart stations that increases interactions between the passengers and the vehicles (Samasti, 2023). These smart stations include cameras, Wi-Fi, ticket sales, card reading and card filling machines.in addition to information screens and a guidance kiosk. Some extra luxuries can be added such bicycle parking as well, phone charging service, and vending machines. However, tracking the number of passengers entering the system can be done via card reading machines that track both the entry and exit. Other studies suggest tracking numbers with cameras or with phones connected to the Wi-Fi provided to the riders.

5. Conclusion

The Internet of things and big data is the next generation improvement to optimize the public transportation systems. And while a plethora of expertise had already discussed similar optimization, with respect to the passengers, the company as well as the environment, this study set to explore the edges of uncharted territory by taking advantage of big data gathered from the public transit system and using it to finesse the service to cater to as many consumers as possible. Applying dynamic scheduling has started to make some traction in the scientific community, where researchers are experimenting with different techniques and comparing it with other traditional methods with real-life data. This technology can be applied to other bus lines, as well as other modes of transportation as well, subways, trams, ferries and even BRT where it can be a potential addition in the metrobus control center (Buran, 2013).

In order to integrate this method into real-life dynamic scheduling, some criteria need to be met. First, acquiring a strong computer that can exert big computational powers to maximize the efficiency of the algorithm in producing results faster. Secondly, to form a system that can incorporate the rush hour effect into the schedule. At the beginning of the day, the produced schedule would depend on the data from the previous day. Then, at every 5-minute intervals, the number of passengers sent from card scanners, and potentially other data gathering methods, is sent to the computer to check for significant deviations from the previous day’s passenger numbers. If the deviations are within a certain acceptable error range, then the schedule does not alter for this time interval. On the other hand, if there are any noticeable changes, then the algorithm is run to alter the schedule.

Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

References

Abdulal, W., & Ramachandram, S. (2011, June). Reliability-aware genetic scheduling algorithm in grid environment. In 2011 International Conference on Communication Systems and Network Technologies (pp. 673-677). IEEE.

Ai, G., Zuo, X., Chen, G., & Wu, B. (2022). Deep reinforcement learning based dynamic optimization of bus timetable. Applied Soft Computing, 131, 109752.

Arifoğulları, Ö., & Alptekin, G. I. (2022). Public Transportation Data Analysis to Estimate Road Status in Metropolitan Areas: The Case of İstanbul. Procedia Computer Science, 210, 12-18.

Baeldung. (06.01.2025). Simulated Annealing in Computer Science. Retrieved from https://www.baeldung.com/cs/simulated-annealing

Buran, B. (2013). Istanbul metrobus system. In Proceedings of the International Conference on Tourism, Transport, and Logistics, Paris (pp. 547-559).

Chuanjiao, S. U. N., Wei, Z. H. O. U., & Yuanqing, W. A. N. G. (2008). Scheduling combination and headway optimization of bus rapid transit. Journal of transportation systems engineering and information technology, 8(5), 61-67.

Databricks. (a.d. 27.12.2024). Comparing Databricks to Apache Spark. Databricks. Retrieved, from https://www.databricks.com/spark/comparing-databricks-to-apache-spark

De Jong, K. A. (1975). An analysis of the behavior of a class of genetic adaptive systems. University of Michigan.

Fox B, McMahon M (1991) Genetic operators for sequencing problems, in Foundations of Genetic Algorithms, G. Rawlins, Ed. Morgan Kaufmann Publishers, San Mateo,CA, Ed. 1991, pp. 284–300.

Freisleben, B., & Merz, P. (1996, September). New genetic local search operators for the traveling salesman problem. In International Conference on Parallel Problem Solving from Nature (pp. 890-899). Berlin, Heidelberg: Springer Berlin Heidelberg.

F. Wang et al., "Millisecond-Scale Real-Time Scheduling of Buses: A Controller-Based Approach," in IEEE Transactions on Intelligent Transportation Systems, vol. 25, no. 7, pp. 7893-7906, July 2024, doi: 10.1109/TITS.2023.3348115.

Gkiotsalitis, K., & Alesiani, F. (2019). Robust timetable optimization for bus lines subject to resource and regulatory constraints. Transportation Research Part E: Logistics and Transportation Review, 128, 30-51.

Holland, J. H. (1992). Genetic algorithms. Scientific american, 267(1), 66-73.

IETT – Istanbul Electric Tram and Tunnel Company. (a.d. 15.10.2024). Route detail: 429A Kirac - Avcilar Metrobus. IETT. [Online: https://iett.istanbul/EnRouteDetail?hkod=429A&routename=kirac-avcilar-metrobus]

Istanbul Metropolitan Municipality. (a.d. 15.03.2024). Hourly Public Transport Data Set. Istanbul Metropolitan Municipality Open Data Portal. [Online: https://data.ibb.gov.tr/en/dataset/hourly-public-transport-data-set]

Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

Jebari, K., & Madiafi, M. (2013). Selection methods for genetic algorithms. International Journal of Emerging Sciences, 3(4), 333-344.

Katoch, S., Chauhan, S. S., & Kumar, V. (2021). A review on genetic algorithm: past, present, and future. Multimedia tools and applications, 80, 8091-8126.

Kirkpatrick, S., Gelatt Jr, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. science, 220(4598), 671-680.

Lee, C. Y. (2003). Entropy-Boltzmann selection in the genetic algorithms. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 33(1), 138-149.

Li, X., Du, H., Ma, H., & Shang, C. (2018). Timetable optimization for single bus line involving fuzzy travel time. Soft Computing, 22, 6981-6994.

Lin, H., & Tang, C. (2021). Intelligent bus operation optimization by integrating cases and data driven based on business chain and enhanced quantum genetic algorithm. IEEE Transactions on Intelligent Transportation Systems, 23(7), 9869-9882.

Lu, K., Liu, J., Zhou, X., & Han, B. (2020). A review of big data applications in urban transit systems. IEEE Transactions on Intelligent Transportation Systems, 22(5), 2535-2552.

Luhua, S., Yin, H., & Xinkai, J. (2011). Study on method of bus service frequency optimal ModelBased on genetic algorithm. Procedia Environmental Sciences, 10, 869-874.

Michalewicz, Z., & Schoenauer, M. (1996). Evolutionary algorithms for constrained parameter optimization problems. Evolutionary computation, 4(1), 1-32.

Mo, S., Bao, Z., Zheng, B., & Peng, Z. (2020). Towards an optimal bus frequency scheduling: When the waiting time matters. IEEE Transactions on Knowledge and Data Engineering, 34(9), 4484-4498.

Moovit (a.d. 11.11.2024). 429A route: Timetables, stops & maps - Avcılar Metrobüs. Moovit. [Available online at: https://moovitapp.com/istanbul-1563/lines/429a/72389532/6567568/en]

Ning, Z., Tao, C., Fei, L., & Haitao, X. (2015). A hybrid heuristic algorithm for the intelligent transportation scheduling problem of the BRT system. Journal of Intelligent Systems, 24(4), 437-448.

Qian, Z., Feng-Lian, W., & Ju, L. (2015). A bus headway optimization model based on genetic taboo algorithm. Journal of Transport Science and Engineering.

Saini, N. (2017). Review of selection methods in genetic algorithms. International Journal of Engineering and Computer Science, 6(12), 22261-22263.

Samasti, M., (2023) Integrated planning of public transportation systems in smart cities–istanbul case study. In International Research and Reviews in Engineering Volume 1, 47-75, Serüven Yayınevi.

Sevim, I., Tekiner-Moğulkoç, H., & Güler, M. G. (2022). Scheduling the vehicles of bus rapid transit systems: a case study. International Transactions in Operational Research, 29(1), 347-371.

Song, C. Y., Wang, H. L., Chen, L., & Niu, X. Q. (2024). An optimized two-phase demand-responsive transit scheduling model considering dynamic demand. IET Intelligent Transport Systems, 18(5), 853-871.

Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

Soon, G. K., Guan, T. T., On, C. K., Alfred, R., & Anthony, P. (2013, November). A comparison on the performance of crossover techniques in video game. In 2013 IEEE international conference on control system, computing and engineering (pp. 493-498). IEEE.

Tang, J., Yang, Y., & Qi, Y. (2018). A hybrid algorithm for urban transit schedule optimization. Physica A: Statistical Mechanics and its Applications, 512, 745-755.

Tang, J., Yang, Y., Hao, W., Liu, F., & Wang, Y. (2020). A data-driven timetable optimization of urban bus line based on multi-objective genetic algorithm. IEEE Transactions on Intelligent Transportation Systems, 22(4), 2417-2429.

Tech Steel & Materials. (06.01.2025) What is Annealing and Why is it Done? Retrieved from https://www.techsteel.net/what-is-annealing-and-why-is-it-done

TUIK, Turkish Statistical Institute: (2024, February 06). The Results of Address Based Population Registration System, 2023 [Press Release]. https://data.tuik.gov.tr/Bulten/Index?p=The-Results-of-Address-Based-Population-Registration-System-2023-49684&dil=2.

TWI - The Welding Institute. (06.01.2025). What is Annealing? Retrieved from https://www.twi-global.com/technical-knowledge/faqs/what-is-annealing

Ushakov, D., Dudukalov, E., Shmatko, L., & Shatila, K. (2022). Artificial Intelligence as a factor of public transportations system development. Transportation Research Procedia, 63, 2401-2408.

Utku, A., & Kaya, S. K. (2022). Multi-layer perceptron based transfer passenger flow prediction in Istanbul transportation system. Decision Making: Applications in Management and Engineering, 5(1), 208-224.

Utku, A., & Kaya, S. K. (2023). New deep learning-based passenger flow prediction model. Transportation research record, 2677(3), 1-17.

Vemuri, N., Tatikonda, V. M., & Thaneeru, N. (2024). Enhancing Public Transit System through AI and IoT. Valley International Journal Digital Library, 1057-1071.

Wei Pu and Yi-Ren Zou, "Using GASA to solve distributed real-time scheduling problems," Proceedings. International Conference on Machine Learning and Cybernetics, Beijing, China, 2002, pp. 958-961 vol.2, doi: 10.1109/ICMLC.2002.1174525.

Welch, T. F., & Widita, A. (2019). Big data in public transportation: a review of sources and methods. Transport reviews, 39(6), 795-818.

Wihartiko, F. D., Buono, A., & Silalahi, B. P. (2017). Integer programming model for optimizing bus timetable using genetic algorithm. In IOP Conference Series: Materials Science and Engineering (Vol. 166, No. 1, p. 012016). IOP Publishing.

ZhongXi, C., QiJie, H., DeBin, F., & ZhenCheng, L. (2019, June). An Intelligent Bus Scheduling System Based On Real-Time Passenger Flow Data. In 2019 International Conference on Robots & Intelligent System (ICRIS) (pp. 174-178). IEEE.

Beykoz Akademi Dergisi, 2025; 13(2), 487-516 ARAŞTIRMA MAKALESİ
Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

Appendices

Appendix A:

t: time interval [0,215], representing 5 minutes within the servicing hours.

s: station [0,16]

C: capacity constant

F: fleet size

A(t,s): rider arriving at time t, and station s.

E(t,s): embarking riders at t,s.

D(t,s): disembarking rider at t,s.

B(t,s): available spots in the bus that arrives at t,s.

R(t,s): remaining riders at t,s.

R(t): the sum of remaining passengers at each time slot or column.

The flowchart of the algorithm by using these notations is given below.

```
1. Import A(t,s), D(t,s) files.
    2. Import libraries, NumPy, pandas, matplotlib, math and random.
       Set up constants, Set T0, alpha, and Tmin, C=50, F= 15, t_max=215, s_max=16
       and P(t,s) array, where P(t+1,s)=A(t+1,s)+R(t,s)
    3. Create zero 2d arrays for I(t,s), B(t,s), R(t,s), E(t,s).
    4. Generate a random initial population ().
    5. Check validity of the solutions, in such a way that the number is buses dispatched
       every 24 t's does not exceed fleet size.
    6. Evaluate the initial solutions via the process below:
        a. for loop t, t=0,t++,t<=tmax:
            i. for loop s,s=0, s++,s<=smax:
                1. if t=0:
                    a. YES: P(t,s}=A(t,s)
                    b. NO: P(t,s}=A(t,s)+R(t-1,s)
                2. If s=0:
                    a. YES, if N(t)>0:
                        i. YES, I(t,0)=1,B(t,0)=C*N(t), continue to step 3
                       ii. NO, I(t,0)=0, B(t,0)=0, E(t,0)=0, R(t,0) = P(t,0),
                           s++, go to step iii
                    b. NO, if t>0:
                        i. YES, I(t,s)=I(t-1,s-1), B(t,s)=B(t-1,s-1) .
                            1. If I(t,s)>0:
                                a. YES, B(t,s) = B(t-1,s-1) + D(t,s),
                                   continue to step 3
                                b. NO, I(t,s)=0, B(t,s)=0, R(t,s)=P(t,s),
                                   E(t,s)=0, s++, go to step 3.a.iii
```

Beykoz Akademi Dergisi, 2025; 13(2), 487-516 ARAŞTIRMA MAKALESİ
Gönderim tarihi: 05.02.2025 Kabul tarihi: 03.09.2025
DOI: 10.14514/beykozad.1634075

```
                        ii.  NO, I(0,s)=0, B(0,s)=0, R(0,s)=P(0,s),E(0,s)=0, s++,
                             go to step 3.a.iii
                    c.  If B(t,s) >= P(t,s):
                        i.   YES (ALL BOARDING), E(t,s)=P(t,s), R(t,s)=0,
                             B(t,s)=B(t,s)-E(t,s), 3.a.s++ go to iii
                        ii.  NO (PARTIAL BOARDING), E(t,s)=B(t,s), R(t,s)=
                             P(t,s)-B(t,s), B(t,s)=B(t,s)-E(t,s), s++ to 3.a.iii
                3.  If s>smax ,  exit loop , t++, go to step 1, otherwise continue the
                    loop
            ii. If t>tmax, exit loop t, otherwise continue the loop
        b.  END, print R(t,s), and sum the values for every row (t value), and save it as
            an 1 dimensional array, and also provide the summation for all elements of
            the array. The higher the value the lower the evaluation.
7.  Select the fit solutions, elites.
8.  Crossover the elite by pairing them at random.
9.  Mutate the offspring.
10. Generate a neighboring solution for the offspring.
11. Check fitness with the evaluation function and accept the new neighboring
    solution if fitness is improved with SA.
12. Update temperature.
13. Repeat steps 11 and 12 till temperature reaches the Tmin threshold.
14. Update population
15. Repeat steps 5 through 14, till max iteration has been reached.
```