

DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

**SIMULATION MODELLING OF A REAL-LIFE
BUS SERVICE SYSTEM AND OPTIMIZATION
OF AVERAGE TRAVEL TIME WITH
SIMULATION INTEGRATED METAHEURISTIC
ALGORITHMS**

by

Doğan Can GÖKMEN

June, 2022

İZMİR

**SIMULATION MODELLING OF A REAL-LIFE
BUS SERVICE SYSTEM AND OPTIMIZATION
OF AVERAGE TRAVEL TIME WITH
SIMULATION INTEGRATED METAHEURISTIC
ALGORITHMS**

**A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül
University In Partial Fulfillment of the Requirements for the Degree
of Master of Science in Industrial Engineering, Industrial
Engineering Program**

by

Doğan Can GÖKMEN

June, 2022

İZMİR

M.Sc THESIS EXAMINATION RESULT FORM

We have read the thesis entitled “**SIMULATION MODELLING OF A REAL-LIFE BUS SERVICE SYSTEM AND OPTIMIZATION OF AVERAGE TRAVEL TIME WITH SIMULATION INTEGRATED METAHEURISTIC ALGORITHMS**” completed by **DOĞAN CAN GÖKMEN** under supervision of **ASSOC. PROF.DR. ŞENER AKPINAR** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science

.....
Assoc. Prof. Dr. ŞENER AKPINAR

Supervisor

.....
Assoc. Prof. Dr. LEYLA DEMİR

.....
Assoc. Prof. Dr. ŞEBNEM AKYOL

(Jury Member)

(Jury Member)

Prof. Dr. Okan FISTIKOĞLU

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENTS

I would like to express my deepest thanks and gratitude to my supervisor Dr. Şener AKPINAR and my ex-supervisor Dr. Özgür YALÇINKAYA for their support, leadership, and for all values that added to me during my MSc education. It would be much harder reach my goals without his knowledge and effort and obviously there would not a thesis which has strong background.

Also, I would like to express my deepest gratitude to my family, who have always supported, valued and encouraged me. The support and belief of my family, which I know will not withhold their support throughout my life, gave me strength.

Doğan Can GÖKMEN

SIMULATION MODELLING OF A REAL-LIFE BUS SERVICE SYSTEM AND OPTIMIZATION OF AVERAGE TRAVEL TIME WITH SIMULATION INTEGRATED METAHEURISTIC ALGORITHMS

ABSTRACT

One of the important problems in the management of transportation systems is the bus scheduling problem. This problem is the problem of determining the most suitable departure schedule for buses with different passenger demands, different departure times and different routes. In this thesis, a stochastic simulation modelling structure has been developed to understand the dynamics of the system, to determine the outputs of the system under different parameters and to be an input to the optimization methods. The goal is to obtain a feasible bus schedule for all buses in the system and to minimize the average travel time of passengers. To obtain feasible bus schedule, four different metaheuristic optimization algorithms are integrated with the simulation model to minimize the average passenger travel time. The first algorithm is obtained by integrating simulation and genetic algorithm (GA), the second algorithm is obtained by integrating simulation and particle swarm (PS), the third algorithm is obtained by integrating simulation with ant colony (AC), and the fourth algorithm is obtained by integrating simulation and artificial bee (ABC).

After developing the simulation models that reflects the system behaviours, the four algorithms mentioned are adapted to the scheduling problem and these algorithms are integrated with the simulation model to produce results by working simultaneously. These optimization algorithms are run to obtain a feasible bus schedule with the best average passenger travel time then the findings are compared and the results are discussed.

Keywords: Bus Scheduling, simulation, simulation optimization, metaheuristics

BİR GERÇEK HAYAT OTOBÜS SERVİS SİSTEMİNİN SİMÜLASYON MODELLEMESİ VE ORTALAMA SEYAHAT SÜRESİNİN SİMÜLASYON ENTEĞRE METASEZGİSEL ALGORİTMALARLA OPTİMİZASYONU

ÖZ

Ulaşım sistemlerinin yönetiminde önemli sorunlardan biri de otobüs çizelgeleme problemidir. Bu problem, farklı yolcu talepleri, farklı hareket saatleri ve farklı güzergâhlara sahip otobüsler için en uygun hareket tarifesinin belirlenmesi problemidir. Bu tezde, sistemin dinamiklerini anlamak, sistemin çıktılarını farklı parametreler altında belirlemek ve optimizasyon yöntemlerine girdi olmak için stokastik bir simülasyon modelleme yapısı geliştirilmiştir. Amaç, sistemdeki tüm otobüsler için uygulanabilir bir otobüs tarifi elde etmek ve yolcuların ortalama seyahat süresini en aza indirmektir. Uygulanabilir bir otobüs tarifi elde etmek ve ortalama yolcu seyahat süresini en aza indirmek için dört farklı metasezgisel optimizasyon algoritması simülasyon modeline entegre edilmiştir. Birinci algoritma simülasyon ve genetik algoritma (GA) entegre edilerek elde edilmiş, ikinci algoritma simülasyon ve parçacık sürüsü (PS) entegre edilerek elde edilmiş, üçüncü algoritma simülasyon ile karınca kolonisi algoritması (AC) entegre edilerek elde edilmiş, dördüncü algoritma ise simülasyon ile elde edilmiştir. simülasyon ve yapay arı (ABC) entegre edilerek elde edilmiştir.

Sistem davranışlarını yansıtan simülasyon modeli elde edildikten sonra, bahsedilen dört algoritma çizelgeleme problemine uyarlanmış ve bu algoritmalar simülasyon modeli ile entegre edilerek eş zamanlı çalışarak sonuçlar elde edilmiştir. Bu optimizasyon algoritmaları, en iyi ortalama yolcu seyahat süresine sahip uygun bir otobüs tarifi elde etmek için çalıştırıldı ve çıktılar birbirleri ile kıyaslanarak sonuçlar tartışıldı.

Anahtar kelimeler: Otobüs Çizelgeleme, eniyileme, benzetim, metasezgiseller

CONTENTS

	Page
M.Sc THESIS EXAMINATION RESULT FORM	ii
ACKNOWLEDGMENTS	iii
ABSTRACT	iv
ÖZ	v
LIST OF FIGURES	ixi
LIST OF TABLES.....	ixi
 CHAPTER ONE - INTRODUCTION	 1
1.1 Background and Motivation	1
1.2 Research Objectives.....	2
1.3 Organization of the Thesis	3
CHAPTER TWO – LITERATURE REVIEW	4
2.1 Bus Timetabling Problem	5
2.2 Bus Scheduling Problem	7
CHAPTER THREE - AN OVERVIEW OF SIMULATION MODELLING AND METAHEURISTICS OPTIMIZATION ALGORITHMS	 11
3.1 Overview of Simulation Modelling	11
3.2 Overview of Metaheuristics Algorithms.....	12
3.2.1 Genetic Algorithm (GA)	13
3.2.2 Ant Colony Algorithm (ACO).....	17
3.2.3 Particle Swarm Optimization (PSO).....	18
3.2.4 Artificial Bee Colony Optimization (ABC).....	21
CHAPTER FOUR – PROBLEM STATEMENT	24
4.1 An Overview to Scheduling and Timetabling.....	24
4.2 A Real-life Bus Service Scheduling Problem	24
 CHAPTER FIVE – SIMULATION MODES OF A REAL BUS SERVICE SYSTEM	 29

SOLUTION OF A REAL BUS SERVICE SYSTEM 47

6.1 Genetic Algorithm (GA)	48
6.1.1 Encoding Process for Genetic Algorithm	48
6.1.2 Flow of Genetic Algorithm for a Real-Life Bus Scheduling Problem.....	49
6.1.3 Procedures of Genetic Algorithm for a Real-Life Bus Scheduling Problem	49
6.1.4 Results of Genetic Algorithm	55
6.2 Ant Colony Optimization (ACO).....	57
6.2.1 Encoding Process for Ant Colony Optimization.....	58
6.2.2 Flow of Ant Colony Optimization for a Real-Life Bus Scheduling Problem	58
6.2.3 Procedures of Ant Colony Optimization for a Real-Life Bus Scheduling Problem.....	59
6.2.4 Results of Ant Colony Optimization	64
6.3 Particle Swarm Optimization (PSO).....	66
6.3.1 Encoding Process for Particle Swarm Optimization	67
6.3.2 Flow of Particle Swarm Optimization for a Real-Life Bus Scheduling Problem.....	67
6.3.3 Procedures of Particle Swarm Optimization for a Real-Life Bus Scheduling Problem	68
6.3.4 Results of Particle Swarm Optimization	73
6.4 Artificial Bee Colony Algorithm (ABC)	74
6.4.1 Encoding Process for Artificial Bee Colony Algorithm	75
6.4.2 Flow of Artificial Bee Colony Optimization for a Real-Life Bus Scheduling Problem.....	75
6.4.3 Procedures of Artificial Bee Colony Algorithm for a Real-Life Bus Scheduling Problem.....	76
6.4.4 Results of Artificial Bee Colony Algorithm	80

CHAPTER SEVEN - CONCLUSIONS.....	82
REFERENCES.....	85
APPENDICES	88



LIST OF FIGURES

	Page
Figure 3. 1 A visual example for crossover operator.....	16
Figure 3. 2 A visual example for mutation operator.....	16
Figure 4. 2 Summary information about routes and buses	28
Figure 4. 2 A visual example for movement logic of buses	28
Figure 5. 1 Created sheet to track passenger movements in the system.....	33
Figure 5. 2 Process flowchart from the drivers's perspective.....	33
Figure 5. 3 Process flowchart from the passenger's perspective.....	33
Figure 5. 4 Observation form for time picking up and dropping off.....	33
Figure 5. 5 An example of input analyzer	34
Figure 5. 6 Output of the Trace element.....	35
Figure 5. 7 Submodels of the model.....	37
Figure 5. 8 Submodels of the model.....	37
Figure 5. 9 Experimental frame of the model.....	37
Figure 5. 10 Passenger arrival model	38
Figure 5. 11 Example create block for passenger arrivals.....	38
Figure 5. 12 Logic of value changing for variables.....	39
Figure 5. 13 Passenger movement model	39
Figure 5. 14 Creating driver model.....	40
Figure 5. 15 Activating the bus model.....	40
Figure 5. 16 Transport between stations model	41
Figure 5. 17 Example of a transport block	41
Figure 5. 18 Example of bus movement.....	42
Figure 5. 19 First and second control logic for buses.....	43

Figure 5. 20 Drop-off logic for buses	44
Figure 5. 21 An example of tally block	45
Figure 5. 22 Output analyzer result for weekday model.....	46
Figure 6. 2 Representation of problem with GA view.....	49
Figure 6. 2 Flowchart of GA.....	49
Figure 6. 3 An example for crossover procedure.....	54
Figure 6. 4 An example for mutation procedure.....	55
Figure 6. 5 Flowchart of ACO	59
Figure 6. 6 Representation of problem with PSO view	67
Figure 6. 7 Flowchart of PSO	68
Figure 6. 8 Flowchart of ABC algorithm.....	76

LIST OF TABLES

	Page
Table 2.1 Summary of literature review on bus timetabling and bus scheduling....	10
Table 4.1 Example real-life schedule for route number 2	28
Table 6. 1 Pseudo codes of genetic algorithm.....	50
Table 6. 2 Parameter combinations for genetic algorithm.....	52
Table 6. 3 Results of genetic algorithm for weekdays model.....	56
Table 6. 4 Results of genetic algorithm for weekend model	57
Table 6. 5 Pseudo codes of ant colony algorithm	60
Table 6. 6 Parameter combinations for ant colony algorithm.....	61
Table 6. 7 Results of ant colony optimization for weekdays model.....	65
Table 6. 8 Results of ant colony optimization for weekdays model.....	66
Table 6. 9 Pseudo codes of particle swarm optimization	69
Table 6. 10 Parameter combinations for particle swarm optimization	70
Table 6. 11 Results of particle swarm optimization for weekdays model.....	73
Table 6. 12 Results of particle swarm optimization for weekend model	74
Table 6. 13 Pseudo codes of artificial bee colony algorithm.....	77
Table 6. 14 Parameter combinations for artificial bee colony algorithm	78
Table 6. 15 Results of artificial bee colony algorithm for weekdays model	81
Table 6. 16 Results of artificial bee colony algorithm for weekend model.....	81
Table 7. 1 Comparison of results for weekdays.....	84
Table 7. 2 Comparison of results for weekend	84

CHAPTER ONE

INTRODUCTION

In this chapter, the background, motivation and objectives of this thesis are explained, and the organization of the thesis is given.

1.1 Background and Motivation

Bus scheduling problem is one of the important problems for transportation management. This problem directly affects operational costs and performance of whole system. For the service sector, customer satisfaction is extremely important and obviously bus schedules directly affect customer satisfaction. In transportation industry buses are commonly used. Both inner city transportation and inner city travelling buses are common vehicles. Therefore, bus scheduling problem is vital for inner city travelling firms and transportation firms.

There are four basic topics in bus scheduling problem; passenger traveling time, passenger waiting time, fullness ratio of bus and cost of operations. Scheduling is a vital tactical decision in transportation industry. Efficiency of the bus systems depends on several key performance indexes which affect customer satisfaction.

When all the issues mentioned are considered, it is clear to see bus scheduling problem plays important role for transportation management and customers. As it is known, the era which we live in requires data collection and proper usage of the data. When transportation sector is considered, it can be said that this sector is affected by data era. In this point, proper usage of know-how and data is vital for the sector and ensures very high competitiveness, profitableness, and sustainability.

Generally, bus scheduling problem is the problem of determining the most suitable departure schedule for buses. This problem also known as *bus timetabling* problem. In the literature there are several keywords for this problem such as *bus planning*, *bus timetabling*, *bus scheduling* and *vehicle scheduling*.

Bus scheduling/timetabling problems has two types. One of them is single depot bus scheduling/timetabling problem. The other is multi-depot bus scheduling/timetabling problem. Single depot scheduling/timetabling problem is the

easiest type of bus scheduling/timetabling problems. In this type all buses in the system start from the same location after completing their trip they come back to the same location. In the other version of bus scheduling/timetabling problem, buses can start from different locations. This type called multi-depot scheduling/timetabling problem.

Generally, there are two main aims for bus scheduling/timetabling problem. These goals can be summarized as; minimizing travelling time or maximizing transported customers while the schedule is feasible for the buses.

This study deals with multi-depot bus scheduling and focuses on increasing customer satisfaction by minimizing average travelling time of customers.

In the literature there are publications related with this problem and commonly mathematical modelling has been used. Since there is not much simulation integrated optimization approach in the literature for a real-life bus scheduling problem this study has been handled. In addition, the fact that the system discussed in this study is a real-life system and this problem is a real-life problem in the transportation industry has formed the main motivation of the study. Also, the stochastic structure of the problem tried to be solved has led us to use optimization methods integrated with simulation modelling.

1.2 Research Objectives

In this thesis, the bus scheduling problem is studied with three main objectives;

- To review the relevant literature.
- To develop a feasible schedule for buses in the problem.
- To develop optimization algorithms for the problem to increase performance of the system.

In order to meet objectives;

- The past projects/studies on bus scheduling and bus timetabling are reviewed and classified according to the problem type, goals, developed model structure(s) and solution approach(es).
- A simulation model has been developed to understand dynamics of bus system and see behaviour of system under different scenarios.

- Developed simulation model has been verified and validated to see whether the model is useful to represent real-life bus service system.
- Average travelling time of passengers is calculated by simulation model and accepted as performance indicator of the bus service system.
- To optimize the performance indicator and to find feasible schedules for the buses metaheuristic algorithms has been used.
- Metaheuristic algorithms integrated to the simulation model. Next, the algorithms run simultaneously with the simulation model. Thus, feasible solutions have been reached and solutions compared and discussed.

1.3 Organization of the Thesis

The organization of this thesis is as follows. In chapter two, a comprehensive literature review has been given. Chapter three comprises an overview of simulation modelling and an overview of metaheuristics algorithms which used in this study such as genetic algorithm, ant colony algorithm, particle swarm optimization and artificial bee colony algorithm. In chapter four, firstly general outline of bus scheduling/timetabling problem, next, the specific problem in the thesis are explained in detail. In chapter five simulation model of the problem and verification/validation studies are given. In chapter six the used algorithms for optimization explained step by step and results are shown. Finally, in chapter seven which is the last chapter, reached results are discussed.

CHAPTER TWO

LITERATURE REVIEW

In transportation industry buses are commonly used. Both inner city transportation and inner city travelling buses are most common vehicles. Therefore, bus scheduling problem is vital for inner city travelling firms and transportation firms.

Bus scheduling problems generally examined in two main titles. Those are bus scheduling problem and bus timetabling problem. The purposes of these problems are generally the same. Determining the appropriate and/or optimal bus frequency regarding to waiting time and/or fullness ratio. In literature there are studies on the problem. Some studies solved this problem by mathematical modelling, some by heuristics or meta-heuristics and some by simulation. This chapter denotes scheduling and timetabling problems and their solutions methods in the literature.

Buses are commonly used in cities for transportation. Therefore, buses highly have importance level in transportation industry. There are four basic topics in the problem: passenger travel time, passengers waiting time, fullness ratio of buses and cost of operations. Scheduling/timetabling is a vital tactical decision in transportation industry. Efficiency of the bus systems depends on several key performance indexes which effect customer satisfaction.

Bus scheduling/timetabling problems has two types. One of them is single depot bus scheduling/timetabling problem. The other is multi-depot bus scheduling/timetabling problem. Single depot scheduling/timetabling problem is the easiest type. In this type all buses in the system start from the same location then complete their trip and come back to the same location. In the second version buses can start from different locations. This type called multi-depot scheduling/timetabling problem.

Despite the words scheduling and timetabling means nearly the same, there is a small difference between them. Generally, the word scheduling means the things that do not need to be repetitive. For instance, when we assign buses to the routes for just one day in a bus system it is a schedule. Despite that when we assign buses to the

routes monthly and if this program repeats itself monthly, it is a timetable. So, we can say the word timetable is used for repetitive works. Bus timetabling and bus scheduling problems share the same purposes such as reducing average travel time, maximizing fullness ratio, running the routes with minimum costs, and minimizing slack time of passengers.

2.1 Bus Timetabling Problem

Klemmt & Stemme (1988) developed a quadratic semi-assignment formulation for the bus timetabling problem. Purpose of the problem was minimizing waiting times of passengers to create useful bus schedule. A model was created in which trips were scheduled one at a time with consideration to transfer synchronization. The goal of the study was minimizing the waiting times of passengers.

Daduna & Voß (1995) proposed a bus timetabling problem to synchronize arrival times at transfer zones with minimizing the waiting time of passengers. Furthermore, the authors considered alternative objectives such as a weighted objective function based on different transfers and the maximum waiting time at a transfer zone. A mathematical model based on quadratic semi-assignment was used to formulate the problem. Due to its complexity, a heuristic was used to compute initial solutions, which were then improved by a simulated annealing algorithm and different versions of tabu search.

Castelli, Pesenti, & Ukovich (2004) developed a solution for a timetabling problem to minimize the time spent in system and the operational costs (based on vehicles usage). The authors developed a mathematical formulation. Since the proposed model had a large number of decision variables, the authors developed a lagrangian relaxation method to solve the generated instances considering a planning period of one hour.

Wang & Shen (2007) showed running time of buses with the limited fuelling. The buses must be re-fuelled after taking several trips and the minimal fuelling time must be satisfied. The main goal of the study was minimizing both number and idle time of buses. To solve the problem, the authors developed an ant colony optimization algorithm which was tested on several examples.

Liebchen, & Stiller (2008) implemented buffer times to generate delay resistant timetables. The authors defined an optimization problem to minimize the expected total weighted delay for planned trips considering headway bounds and fleet size constraints. To solve the problem, the authors developed a heuristic algorithm which is capable of obtaining optimal solutions for problems that can be modelled through series-parallel graphs. Although this was implemented in periodic timetables for railway systems, the main idea of implementing buffer times can also be used for general transit networks with aperiodic timetables.

Li, Lam, Wong, & Sumalee (2010) suggested a model for the periodic timetabling problem in a multi-modal system in order to maximize the total passenger's net utility. The authors developed binary mathematical programming problem with supply-demand balance constraints and implemented heuristic procedures to solve small examples leading to headway values that improve the user's net utility.

Hassold & Ceder (2013) presented a formulation for the multi-depot bus timetabling problem. The aim was minimizing bus operating costs based on buses usage ratio and number of empty seats. This study aimed creating unique-flex bus timetables using by partial timetables and has two assumptions. These were buses can be substituted, and substituted bus has sufficient capacity. The authors tested their model on two bi-directional lines of the transit network in Auckland, New Zealand. The model gave better solution regarding operating costs to multi-depot bus timetabling problem. Further, the authors analysed the effect of the solution on the level of service, in terms of waiting times and empty seats.

Wihartiko, Buono, & Silalahi (2017) developed an integer programming formulation modified with genetic algorithm to solve a bus timetabling problem. Modification was placed in the chromosomes design, initial population recovery technique, chromosomes reconstruction and chromosomes extermination on specific generation. The goal was to determine optimal number of trips under the passenger demand.

Shariat & Amiripour (2019) took over bus timetabling problem in terms of stochastic demand. In their model passenger demand was changing during the day.

They formulated mathematical model and developed simulation algorithm to get an optimal schedule under stochastic passenger demand. A distribution function was used instead of a deterministic demand function. The goal of the study was maximizing the temporal coverage of bus network.

2.2 Bus Scheduling Problem

Saha (1970) developed an algorithm for bus scheduling problem. A linear mathematical model used for solution. Purpose of the study was to determine minimum required number of buses and minimum required crew. Different mathematical models prepared and solved.

Baita, Pesenti, Ukovich, & Favaretto (2000) suggested several solutions to solve a bus scheduling problem considering the following practical elements; empty buses are allowed, fuelling is possible only when trip is completed and bus waits until the next scheduled trip. The aim of study was minimizing fleet size, minimizing required bus routes (lines), minimizing waiting time of buses for the next scheduled trip and minimizing cost of buses which is running empty. The authors created three solution approaches and compared them. In the first approach there was a mathematical formulation with a solver considering a weighted objective function, in the second one a logic programming method considering a lexicographic order for the different criteria and in the third one there was a multi-objective genetic algorithm to find the optimal solution.

Palma & Lindsey (2001) assumed that each passenger has an ideal boarding time and variable delay cost from traveling earlier or later. Authors suggested an analytic model which had two aims to solve demand allocation problem and problem of minimizing costs. Demand allocation problem was solved to assign individuals to line runs and minimizing problem was solved to set departure times in such a way as to minimize the passengers' total delay costs.

Kim, Kim, & Park (2012) scheduled a school bus which contains trips for every school with fixed time period. Study aimed to optimize bus schedules to serve all the given trips considering the school time period. Different scenarios were developed and a mathematical model developed for every scenarios. Every model

solved with different algorithms such as branch and bound, construction algorithms. They recommended multi integer programming model for the problems. The study showed that branch and bound algorithm is useful for small fleet problems but on the other hand for big scale problems heuristics can be useful.

Wei, Sun, & Jin (2012) highlighted that the bus scheduling problem is a trending topic in the urban transport sector and will gain more importance in the future. In the study, optimal bus schedules were calculated with ant colony algorithm. The main purpose of the study was to reduce operating costs. In this context, it has been observed that the proposed optimization model is more efficient than real-life results. The study was conducted on a multi-depot system and considering fuelling, bus capacity and variable travel times.

Zhoucong, Peijia, Jing, & Liping (2013) formed a method based on clustering algorithm to make short planning periods with low variability of travel times. Single depot bus scheduling problem was solved. The authors executed the proposed method in a real-life problem which was in Shanghai. Their purpose was dividing day with clustering method to determine the bus schedule with the minimum operating cost. The results showed that a proper division of the day leads to a more accurate representation of vehicles' usage costs.

Pekel & Kara (2015) developed a discrete-event simulation model for a real-life bus scheduling problem. Arena simulation model was used in the study. Different scenarios created and simulated, then compared. Goal of the study was creating schedules which meet passengers demand. At the end of the study the best scenario had found. On the other hand according to absence of a cost function was a disadvantage. To improve the study, cost function should be added.

Wei, Chen, Sun, & Zhu (2015) tried to create bus schedules with the maximum utilization of fleet buses. In contrast to traditional single-depot, their study contains coupled schedule for buses. Because of real factors there was delay for buses and to model uncertain delay time fuzzy numbers were used. They used constraint programming in the study. This problem contained fuzzy travel times and solved with partial swarm optimization algorithm and group search optimization.

Öner, Gultekin, & Koç (2020) studied a real-life problem about airport shuttle system. They tried to find optimal schedule for shuttles which takes passenger from an airport and drop at a stop. In the study, a mixed integer mathematical programming and metaheuristics approach have been applied and solutions were compared. Furthermore, they investigated effect of parameters on service quality level such as waiting time, transportation fee, bus capacity and travelling time.

Gkiotsalitis (2020) proposed a heuristic for the bus scheduling problem. While addressing the problem, he considered passenger demand and bus capacity. This study, including a simulation optimization method, focused on a real-life problem. The study was conducted with real-life data on a bus line in Singapore, and the average passenger travel time was reduced by approximately 20%.

The briefly introduced studies in the above are summarized in Table 2.1.

Table 2.1 Summary of literature review on bus timetabling and bus scheduling

#	Author(s)	Problem Type	Performance Criteria	Solution Approach
1	J.L. Saha (1970)	Bus scheduling	Number of buses, Number of crew	Mathematical Model
2	Klemmt & Stemme (1988)	Bus timetabling	Waiting time of passengers	Mathematical Model
3	Daduna & Voss (1995)	Bus timetabling	Waiting time of passengers	Mathematical Model, Metaheuristics
4	Baita et Al. (2000)	Bus scheduling	Fleet size, required bus lines, operating cost, waiting time of buses	Mathematical Model- Metaheuristics
5	Palma & Lindsley (2001)	Bus scheduling	Operating costs	Mathematical Model

Table 2.1 Continues

6	Castelli et al. (2004)	Bus timetabling	Time spent in system, usage of buses	Mathematical Model
7	Wang & Shen (2007)	Bus timetabling	Number of buses, idle time of buses	Metaheuristics
8	Mohaymany & Amiripour (2009)	Bus timetabling	Service Time	Mathematical Model, Simulation
9	Li et al. (2010)	Bus timetabling	Passenger's net utility	Mathematical Model, Heuristic
10	Liebchen & Stiller (2012)	Bus timetabling	Total excepted delay time of buses	Heuristics
11	Wei et al. (2012)	Bus scheduling	Minimizing operational cost	Metaheuristics
12	Kim et al. (2012)	Bus scheduling	Number of serviced points	Mathematical Model
13	Zhoucong et al. (2013)	Bus scheduling	Variability of travel times, Operating cost	Clustering
14	Hassold & Ceder (2014)	Bus timetabling	Usage of buses, number of empty seats	Mathematical Model
15	Pekel & Kara (2015)	Bus scheduling	Number of serviced passenger	Simulation
16	Wei et al. (2015)	Bus scheduling	Utilization of buses	Fuzzy Logic, Constraint Programming, Heuristics
17	Wihartiko et al. (2017)	Bus timetabling	Number of trips	Mathematical Model, Metaheuristics
18	Öner et al. (2020)	Bus scheduling	Maximizing operation revenue	Mixed integer programming, Heuristics
19	Gkiotsalitis (2020)	Bus scheduling	Minimizing average travelling time	Simulation, Heuristics

CHAPTER THREE

AN OVERVIEW OF SIMULATION MODELLING AND METAHEURISTICS ALGORITHMS

3.1 Overview of Simulation Modelling

Simulation is a tool that provides an understanding of the dynamics of a system, shows the properties of the system and how it works under different parameters. System simulation is seen as a very powerful tool. It is a unique tool for seeing the future and doing what-if analysis, especially in systems where it is very costly or risky to make changes. Today, it is very important to collect data and to provide information using the collected data and to stay one step ahead of the competition. Therefore, the importance of simulation modelling is undoubtedly increasing. Also, many problems in real-life are dynamic and stochastic. Simulation modelling approach comes to the fore when problems cannot be solved with mathematical approaches and system structures cannot be understood. With the simulation approach, less costly solutions can be found in a shorter time. Also, this approach is an excellent tool not only for finding solutions but also for analysing different scenarios/strategies.

Simulation can be defined as the process of designing a model of a real system conducting experiments with this model for the purpose of understanding the behaviour of the system (Pegden, Shannan, & Sadowski, 1995). Simulation modelling approach can be useful in scenarios below occur.

- When there is no mathematical expression for the problem or mathematical expression complex to get for the problem.
- When the system is not suitable for experiment or when experiment cost is very high and risky.
- When there is no analytical solution for mathematical model.
- When analytical solution exists, but it takes lots of time and it is too complex.

To have better understanding of simulation term, model and system terms must be explained. These two terms are main components of simulation. The word system

can be defined as the coming together of a group of elements or subsystems that are related to each other and serve the same purpose. In the other hand model can be described as a different form of systems or objects in a digital way or in a computer language.

Simulation models can be classified in many ways but mainly they can be classified as iconic and symbolic models. While iconic models look in some sense, like the real system, symbolic models are usually run on a computer, and indeed, in most cases simulation would be of no practical value without computers (Pegden, Shannan, & Sadowski, 1995).

If the parameters showing the state of the system are constantly changing over time depending on a function, this system is called continuous system. However, if these parameters are triggered by an event occurring in the system and do not change continuously with time, this system is called discrete event system. The simulation model in this study is an example of discrete event system simulation. Discrete event system simulation can be used in many areas. These clauses summarize the usage area of discrete event system simulation;

- Manufacturing Problems
- Logistics and Supply Chain Problems
- Scheduling Problems
- Production Planning Problems
- Manpower Planning Problems
- Strategic Management Problems
- Inventory Problems
- Military and Governance Problems

3.2 Overview of Metaheuristics Algorithms

Optimization can be defined as the process of finding the best value as a solution to a problem. The problem to be solved may be continuous or discrete. There are many optimization methods. However, in general, it is possible to collect optimization methods under two headings as mathematical optimization and heuristic

optimization methods. The result sought in an optimization problem may be minimization or maximization of the function. Optimization methods are used for both minimization and maximization problems. If the problem can be presented as a mathematical function, a mathematical optimization modelling approach will be appropriate for the solution. However, heuristic methods are used if the function cannot be solved with an analytical point of view or if it cannot give a suitable solution in the desired time interval (Rardin, 2000).

In particular, heuristic optimization approaches are used in solving NP-hard problems. While mathematical optimization gives optimum results, heuristic optimization approaches produce near optimum results. For very large and very complex optimization problems, heuristic approaches are highly preferred. Heuristic approaches generally give the near optimum result in an appropriate time. It is seen that heuristic approaches are frequently used for real-life optimization problems. Some heuristic algorithms can only be used for limited type of problems. On the other hand, there are heuristic algorithms used in all kinds of optimization problems. These algorithms are called metaheuristics algorithms. Metaheuristic algorithms are optimization algorithms that give appropriate results for all types of optimization problems in suitable times. It can be said that both heuristic optimization algorithms and metaheuristic optimization algorithms are inspired by nature, biology, swarm behaviour and physics. In this thesis, metaheuristic algorithms are used. The used algorithms are Genetic Algorithm (GA), Particle Swarm Optimization Algorithm (PSO), Ant Colony Algorithm (ACO) and Artificial Bee Algorithm (ABC). These algorithms are used for many types of problems such as scheduling, routing, and planning in literature.

3.2.1 Genetic Algorithm

GA is a frequently used metaheuristic optimization algorithm. This algorithm is one of the most prominent among the Evolutionary Algorithms and has a wide range of use in many problem types. The GA was first introduced by Holland (Holland, 1975) and became famous with the work of David Goldberg. GA is an algorithm inspired by the biological system and its basic teaching is to continue the generation of individuals with the best adaptation ability. GA is frequently used for scheduling,

routing, assignment problems and has been successfully adapted for combinatorial optimization problems. It can be said that GA highly successful optimization algorithm to find the optimal or near optimal solution in a reasonable time for highly complex problems.

Since GA is a heuristic, the algorithm starts searching solution space from random points. Every point in solution space is a candidate solution for the optimum solution. In the concept of GA every solution called chromosome. Chromosomes consist of genes. Genes represent a part of solution. Position of a gene called locus and value of a gene called allele. A special operator which is called crossover combines chromosomes and generates offspring. Generated offsprings mutated by mutation operator and turns into a new chromosome. This process goes like this until desired fitness values caught or stopping criteria is met.

Main steps of GA as it follow;

1. Parameter definition (Population Size, Crossover Rate, Mutation Rate)
2. Generating of initial random solutions
3. Objective function evaluation (Fitness Value)
4. Parent selection
5. Crossover process (Offspring)
6. Mutation process
7. Control of stopping criteria

First of all, the parameters must be defined. These parameters have significant effect on algorithm. Population size defines number of random solutions. In the literature it can be seen that as population size increases, the fitness value of solutions also improves. On the other hand there is a disadvantage usage of high population size. High population size requires more time for a final solution. With the crossover operator the algorithm generates new solutions (children) from selected parent (mother and father) solutions.

Crossover rate is directly affecting number of individuals on new generation, and it can be described as expected ratio of parents. This rate can be defined high or low. While high Crossover rate usage allows exploring more solution space, low Crossover rate usage can trigger bad fitness values due to searching of less solution space.

Mutation ratio can be defined as the ratio for number of altered genes for new generations. If the ratio is too high randomness for chromosomes will be high and solution space will be explored well. However, usage of high ratio causes slower algorithm. On the other hand too low mutation ratio causes loose learning from history ability for the algorithm and causes bad fitness values.

After parameter definition initial solutions must be generated. Number of initial solutions equals to population size. Then, fitness values catch, and evaluation of fitness values process starts. Evaluation process is important to see the best fitness value and the best chromosome. It can be said that evaluation of fitness values process is an input for parent selection. The best fitness value depends on problem type. For minimizations problems lowest value is the best fitness value. For the maximization problems the situation is on opposite.

Parent selection is a process to select parents to generate new children for next generations. There is more than one parent selection method in the literature. Some of them are roulette wheel selection, ranking selection, and tournament selection. In this study ranking selection is used for parent selection. Fitness values are sorted from the best to the worst in this selection method. Thus, algorithm knows which solutions are good to generate children for next generations. In parent selection process the number of selected solutions as parent must be even. The reason of that is one of the parent is mother and the other is father. Two solutions generate one new solution.

Chromosomes are combined with crossover operator. Crossover is one of main operator of GA. This operator combines genes of parent chromosomes in order to generate offsprings. Like parent selection there are many methods for crossover operator in the literature. In this study single point crossover method is used. In this

method where the number of genes in a chromosome is L (which also represent length of a chromosome) a cut point is determined randomly. Cut point must be more than 1 and must be less than length of chromosome. Every parent has a random cut point. Then with the genes of the mother chromosome before cut point are combined with the genes of the father chromosome after cut point. Thus, offsprings are generated. An example is given in figure 3.1.

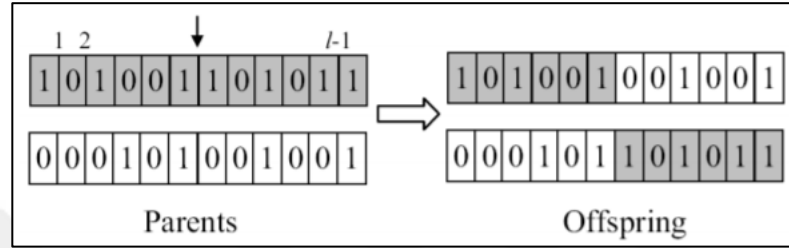


Figure 3.1 A visual example for crossover operator (Yalçinkaya, 2010)

In order to ensure genetic diversity, there is mutation operator. Mutation operator is a special designed operator to change allele of a gene. Also, mutation operator can be applied more than one gene. This operator selects a gene or genes from a chromosome and alter allele of them randomly. Thus, genetic diversity is ensured. An example is given in figure 3.2.

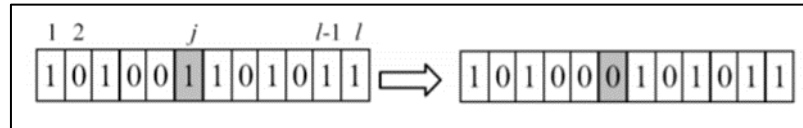


Figure 3.2 A visual example for mutation operator (Yalçinkaya, 2010)

After mutated chromosomes generated, the process continues with catching fitness values of new chromosomes and an iteration finish. Iterations repeats itself until stopping criteria is met. Algorithm runs until defined iteration number. Also there are several techniques to stop algorithm.

Low number of iterations causes low number of generations which brings worse fitness values due to searching tight solution space. On opposite, too high iteration numbers cause over running of algorithm and it takes time. Therefore, in this study there is a restriction for stopping criteria. When the best fitness values do not improve

itself after particular number of iterations the algorithm stops. Thus, disadvantages of too low and too high iteration number has been eliminated.

3.2.2 Ant Colony Algorithm

ACO Algorithm is also broadly used algorithm in the literature for complex optimization problems. Algorithm first introduced in Dorigo (1992). First shape of this algorithm called Ant System. Following years with other studies on the algorithm, general framework has been determined and called ACO. In the literature, ACO algorithm frequently used for shortest pathway problem. Nevertheless, it can be said that ACO algorithm is useful for scheduling problems. This optimization algorithm inspired by behaviours of ants in the nature. Ants try to find shortest way while searching food source in the nature. An ant in the system leaves a chemical liquid called pheromone on its paths in order to communicate with other ants in the system. While the density of this liquid is inversely proportional to the length of the path, it is directly proportional to the quality of the food source found. With pheromone ants communicate each other's and find the shortest way to reach food source. Thus, swarm intelligence is obtained, and the algorithm learns from the past.

Main steps of ACO Algorithm as it follow;

1. Parameter definition (Population Size, α , β , ρ)
2. Generating of initial random solutions
3. Generating initial pheromones and initial probabilities
4. Catching fitness values
5. Fitness value evaluation process
6. Updating pheromones and updating probabilities
7. Control of stopping criteria

First of all, the parameters must be defined. These parameters have significant effect on the algorithm. Population size refers to number of ants in the system. In the other hand α , β , ρ parameters are extremely important. The parameter α reflects importance of quantity of pheromones. If this parameter is too high, ants in the system will always follow where pheromone is high. But in this situation new solutions cannot be explored well due to searching tight solution space. The

parameter β reflects to power of stochastic searching. High β parameter causes lack of learning from the past for ants in the system. The parameter ρ is a coefficient for evaporation of pheromones. Pheromones between arcs decreased using by this evaporation coefficient. While ρ is too high it is hard to find optimum solution. After parameter definition, virtual ants generate by the algorithm, as population size. After defining the parameters, the algorithm generates the population of ants. Each ant represents a solution in solution space and each solution has a fitness value. One of main step of ACO algorithm is pheromone update step. The amount of pheromone released by the ants is directly proportional to the quality of the solution. The better the fitness value reached, the greater the amount of pheromone released. If the amount of pheromone belonging to a solution is high, the ants in the system will have a high probability of choosing that solution in the next iteration. From this point of view, it can be said that high pheromone amount directly affects the next iterations and swarm intelligence. In this thesis roulette wheel selection method is used for determining next node of ants. Thus, the probability of selecting nodes with high probability values is increased. In addition to this, Elite Ant method is used. The ant which has the best fitness value called Elite Ant in the system. After all these steps, an iteration in the algorithm is completed. These iterations continue until the stopping criterion is met. The stopping criterion is usually stated as the number of iterations.

3.2.3 Particle Swarm Optimization

PSO is a metaheuristic optimization algorithm inspired by the behaviour of birds and fish in the nature. It is frequently used in solving combinatorial optimization problems. This algorithm prioritizes swarm intelligence. It has been observed in the literature that this algorithm, which was developed by Kennedy and Eberhart in 1995, gives successful results in NP-hard problems. In the PSO algorithm, each iteration is affected by the previous iteration. Thus, the particles in the algorithm are provided to learn from the past and swarm intelligence is used to reach the optimum (Kennedy & Eberhart, 1995). PSO algorithm is designed for continuous problems. However, it can be adapted for discrete problems. The algorithm starts with generating random solutions. Each random solution is called a particle. All the particles come together

to form the swarm. Each particle in the swarm has its own starting point. This starting point is called the position in the algorithm.

At the same time, each particle has its own initial velocity. These velocities are updated throughout the iterations, and the updated velocities lead to the positions being updated. The fact that each particle is in a different position reflects a different solution.

In addition to these, there are other important components of the algorithm. These components are the social coefficient and the cognitive coefficient. These parameters have a significant effect on the success of the algorithm.

Main steps of PSO Algorithm as it follow.

1. Parameter definition (Population Size, w , $c1$, $c2$)
2. Generating of initial random solutions and initial velocities
3. Catching fitness values
4. Velocity update
5. Position update
6. Control of stopping criteria

Parameters have significant effect on success of the algorithm. The first step of the algorithm is parameter definition. The population size determines the number of solutions in the swarm. Keeping the population size too small will cause the algorithm not to explore the solution space well. Therefore, the population size should be chosen appropriately. w reflects the inertia coefficient. This coefficient can be summarized as the tendency of the particle to maintain its position. The social coefficient is denoted by $c1$. This coefficient particle is characterized as the desire to approach the position of the best particle in the swarm. In addition, it expresses the cognitive coefficient with the $c2$ coefficient. This coefficient is the desire to return to the best position of the particle. All these parameters influence the algorithm. If the inertia coefficient (w) is too high, the particle will not change its location and the solution space will not be scanned well. Likewise, if the cognitive coefficient is too

high, it will trigger the particle's desire to constantly return to its best position and increase the risk of being caught in the local minimum.

In the second step of the algorithm, random solutions are generated. The number of random solutions is equal to the population size. Each random solution is called a particle. Every particle has an initial velocity. There are different approaches in the literature regarding initial velocities, but in this thesis, the initial velocity of each particle was determined as zero. Every particle has fitness value. At each iteration, the fitness values of the particles are calculated. In addition to these, the best fitness value in current iteration, the best fitness value has been captured in previous iterations so far and the individual best value of each particle are recorded in the algorithm. These fitness values are very important for the functioning of the algorithm. An iteration moves with the fitness values calculated in the previous iteration.

The particle velocities are updated in each iteration of the algorithm. This velocity update is obtained with the help of the following formula (3.1).

$$V_{ik+1} = wV_{ik} + c_1r_1(Pbest_{ik} - X_{ik}) + c_2r_2(Gbest^k - X_{ik}) \quad (3.1)$$

V_{ik+1} determines velocity vector calculated for $k+1$ iterations of the i th particle. w , c_1 and c_2 explained above. r_1 and r_2 are random number between 0 and 1. While $Pbest_{ik}$ reflects the best fitness value obtained for particle so far, $Gbest$ reflects the best fitness value obtained for whole population so far. X_{ik} determines the position of i th particle at k th iteration. With this formula velocity vector calculated for each iteration. After updating the velocity vectors, the positions of the particles are updated. The new position of a particle is obtained by adding the velocity vector to its current position. The new positions of the particles are obtained by the following formula (3.2).

$$X_{ik+1} = X_{ik} + V_{ik+1} \quad (3.2)$$

X_{ik+1} defines the new position of the particle i in the $k+1$ iteration, while X_{ik} defines the current position of particle i . V_{ik+1} shows the velocity vector of particle i

in the $k+1$ iteration. After all these steps, an iteration in the algorithm is completed. These iterations continue until the stopping criterion is met. The stopping criterion is usually stated as the number of iterations.

3.2.4 Artificial Bee Colony Optimization

ABC algorithm is a metaheuristic optimization algorithm based on swarm intelligence inspired by the behaviour of bees in the nature. This algorithm was developed by Karaboğa in 2007. The basis of the algorithm is the bees looking for a food source in the nature and their communication abilities. Bees in the nature leave their hives in order to find a food source, and at first they randomly search for a food source. When a food source is found, they convey the presence of a food source with different dance figures and the quality of this food source to other bees. Thus, all bees in the colony tend to the point where food sources are more abundant and swarm intelligence is used (Karaboğa & Basturk, 2007).

The algorithm consists of three main phases. These phases are respectively the worker bee phase, the onlooker bee phase and the scout bee phase. In the worker bee phase, bees randomly discover food sources and pass on the information they receive from these discoveries. The transmitted information is used by the onlooker bees and they search in the relevant source area. This phase is called the onlooker bee phase. In the scout bee phase, which is the last phase of the algorithm, scout bees search for new food sources independently of the information from the worker bees. The food resources included in the algorithm are not infinite. A worker bee is assigned to each food source. When the food source is depleted, the worker bee turns into a scout bee and starts to search for the new food source randomly. While the local search is completed with the onlooker phase, the global search is completed with the scout bee phase.

Main steps of ABC Algorithm as it follow;

1. Population size creation
2. Catching fitness values
3. Worker bee phase
4. Scout bee phase

5. Onlooker bee phase
6. Control of stopping criteria

The population consists of worker and onlooker bees. The number of worker and onlooker bees is equal. At the beginning of the algorithm, solutions are randomly generated, and each solution represents a food source. Only one worker bee was assigned to each food source. Like the other algorithms described in this thesis, the effect of the number of worker bees on the success of the algorithm cannot be denied. Using a very low number of worker bees causes the algorithm not to search the solution space well enough. Initial solutions are obtained via following formula (3.3).

$$X_{ij} = X_{jmin} + rand(0,1)(X_{jmax} - X_{jmin}) \quad (3.3)$$

Index j reflects number of variable while index i reflects number of worker bee. In addition to this X_{jmin} and X_{jmax} reflects the bound of problem. After the initial solutions are produced, fitness values are calculated. Since the problem in this thesis is a minimization problem, the lowest fitness value was accepted as the best. In the worker bee phase, the worker bees are sent to a food source. These food sources represent solutions in the solution space. In this phase, worker bees search for neighbourliness. Each worker bee searches for a neighbouring solution and calculates the fitness value in this neighbouring solution. If the fitness value in the neighbouring solution is better than the current fitness value, the worker bees leave the current source and head towards the neighbouring source. This selection is called the greedy selection method. The new food source adjacent to the current solution is created by the following equation (3.4).

$$V_{ij} = X_{ij} + \emptyset_{ij}(X_{ij} - X_{kj}) \quad (3.4)$$

With this equation, a new food source V_i is produced adjacent to each X_i source. Here, j represents the parameters that make up the source and its value is the total number of parameters. It is a randomly generated integer. k denotes food sources, its value is a randomly generated number in the range $[-1,1]$. One of the points to be considered at this stage is the failure counter. If a worker bee cannot obtain a better

fitness value when it searches for a neighbouring source, the failure counter is increased by one for the current food source, but otherwise the counter is reset.

In the onlooker bee phase, the worker bees return to the hive and transfer the quality of the source they find to the onlooker bees. This phase is the phase in which learning is provided in the algorithm. Onlooker bees choose a source for themselves according to the information transmitted from the worker bees. The roulette wheel method is used in the selection of this resource. The probability of selecting each resource is calculated using the formula below (3.5).

$$P_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n} \quad (3.5)$$

SN denotes number of bees and P_i denotes probability and fit_i denotes fitness value for source i in this equation. The higher the quality of the solution in the source, the higher the probability of being selected. A number is set for maximum failure. If the failure value of a source has reached the maximum number, the worker bees in that source turn into scout bees and start searching for a random source. A global search is made with the scout bee phase. The loop continues until the specified maximum number of iterations is reached. In each iteration, the best solution in the iteration and the best solution obtained so far are kept in memory.

CHAPTER FOUR

PROBLEM STATEMENT

4.1 An Overview to Scheduling and Timetabling

Scheduling is the problem of assigning resources to tasks in the production of a product or a service and finding the optimal balance of which resource will do which work at what time. Scheduling is a frequently used method in resource planning. With the scheduling method, it is ensured that the tasks are completed before the due dates or delays are minimized. Scheduling problems are frequently encountered in the literature. The most common scheduling problem is the production scheduling problem. This problem deals with which jobs should be done on which machines and at what times. In addition, scheduling problems are frequently encountered in project management. In fact, it can be said that scheduling problems are frequently encountered in daily life. There are scheduling problems that directly affect daily life, such as the departure and landing times of a plane. From this point of view, it can be said that scheduling problems are of great importance for both industry and social life.

Timetabling, on the other hand, has a very close meaning to the word scheduling. The meaning of the word timetabling can be summarized as the act of scheduling something to happen or do something at a particular time. Schedules of public transportation vehicles such as train, bus, metro, tram, ferry are called timetable.

The general objectives in scheduling and timetabling problems can be summarized as being able to produce and to serve in a shorter time. In general, scheduling problems are complex problems. Since the parameters in the scheduling and timetabling problems are related to each other, they are also difficult to solve. Also, scheduling problems are often real-world problems. Many difficulties are encountered in modelling real-world problems. For instance, there are problems such as collecting and analysing the necessary data or the fact that the collected data does not reflect the truth. In addition, scheduling problems are one of the most important issues in combinatorial optimization, and most real-life scheduling problems are NP-hard. Therefore, heuristic methods are generally used instead of exact methods.

4.2 A Real-life Bus Service Scheduling Problem

Today, with the effect of increasing population, the demand for public transportation vehicles has increased. Therefore, customer satisfaction is an extremely important criterion in public transportation vehicles. In addition, minimizing operational costs is extremely important. Bus transportation has a very large share in the transportation sector. Today, buses are frequently used for transportation within the city. In addition, buses and other public transport vehicles have a significant impact on both social life and environment. Therefore, the movement of buses in the optimal schedule affects both operational costs and customer satisfaction. One of the most important key performance indicators in the transportation sector is average travel time. Average travel time is the sum of the time a passenger waits to receive service and the average time spent receiving service. While the average travel time directly affects customer satisfaction, it indirectly affects operating costs and gross income of a transport company. From this point of view, it can be said that bus schedules are vital for both social life and industry.

In this study, a feasible schedule is proposed for a real-life bus system. Simulation optimization method is used while generating a feasible schedule. First, the current situation is modelled in ARENA simulation software, then it is optimized with Genetic Algorithm, Ant Colony Algorithm, Particle Swarm Optimization and Artificial Bee Colony Algorithms. The problem addressed in this study is a real-life problem in which the passenger transportation system of a shopping center in Izmir is modelled and optimized. This problem type is not frequently encountered in the literature. The transport system includes 134 bus stops, 11 bus routes and 20 different buses. More than one bus can share a route in the system. In addition, a bus stop serves multiple routes and multiple buses. This complicates the nature of the problem. The summary information of the system is given in figure 4.1.

Route	Platform No	Bus Number	Vehile Number	Driver Number	Plaque	Starting Time	Ending Time	Double Trip Num	Trip Number	Total Stop in Route	Daily Working Hours	Daily Working Minutes
Evka-5	1	2	1	1	35 KE 8135	11:00	20:30	10	20	16	09:30	570,00
			2	2	35 AED 896	11:30	21:00	10	20	16	09:30	570,00
Evka-2	2	2	1	3	35 EL 676	10:30	21:00	10	20	17	10:30	630
			2	4	35 KE 7391	11:30	21:30	10	20	17	10:00	600
Egekent	3	4	1	5	35 AY 2092	10:00	21:00	10	20	19	11:00	660
			2	6	35 HU 486	10:30	21:15	10	20	19	10:45	645
			3	7	35 MIU 50	10:45	21:30	10	20	19	10:45	645
			4	8	35 EC 3016	11:00	21:45	10	20	19	10:45	645
Tersane	4	3	1	9	35 KE 8279	10:30	21:15	9	18	19	10:45	645
			2	10	35 PLG 90	10:50	21:30	9	18	19	10:40	640
			3	11	35 AGK 759	11:10	21:50	9	18	19	10:40	640
Yeni Girne	5	2	1	12	35 KE 7259	11:00	21:30	11	22	14	10:30	630
			2	13	35 PIP 99	11:30	21:00	10	20	14	09:30	570
Bostanlı	6	2	1	14	35 AT 2447	10:30	20:00	10	20	11	09:30	570
			2	15	35 R 4645	11:00	21:00	10	20	11	10:00	600
Esin Sitesi	7	1	1	16	35 KE 7359	11:00	20:30	10	20	10	09:30	570
Bahçelievler	8	1	1	17	35 KE 8384	11:00	21:30	9	18	21	10:30	630
Harmandalı	9	1	1	18	35 KE 5927	10:00	21:30	10	20	19	11:30	690
Elit Sitesi	10	1	1	19	35 KE 7468	11:00	20:00	9	18	20	09:00	540
Şemikler	11	1	1	20	35 S 95862	12:00	17:05	6	12	16	05:05	305

Figure 4.1 Summary information about routes and buses

In addition to all this information, the transportation system in question works as follows;

- There are 20 busses and 11 different routers in the system.
- The capacities of the buses in the system are fixed and 27 seats. No standing passengers are allowed during traveling. Therefore, if the number of passengers in the bus is 27, no more passengers can be taken.
- Each route has a start and an end stop. Each departure between them is called a trip.
- Each bus has a predefined number of trips it has to make. Buses cannot leave the system before this trip number is reached.
- Buses travel between the start bus stop and the shopping center which is called end bus stop.
- The other stops between the start and end stops are called road bus stops.
- Each bus stop has different demands. These demands vary significantly on weekdays and weekends. In addition, the number of passengers arriving at a bus stop in the morning, noon and evening hours during the day shows a significant change.
- Each bus has a station time for its start and end bus stops. This station time indicates the time at which the relevant bus should be at the relevant stop. Delays are allowed in the real-life system.
- The departure times, start point, end point and the visited stops of each route are determined.
- There is an existing timetable for each bus and the buses are obliged to follow this timetable. An example schedule given in Table 4.1.
- If the bus has started from a starting station, it only collects passengers and arrives at the shopping center stop and drops them there. However, if the bus has started from a shopping center, it collects the passengers from the shopping center and drops them in order at the other stops. It only picks up passengers from the shopping center. This logic given in figure 4.2.

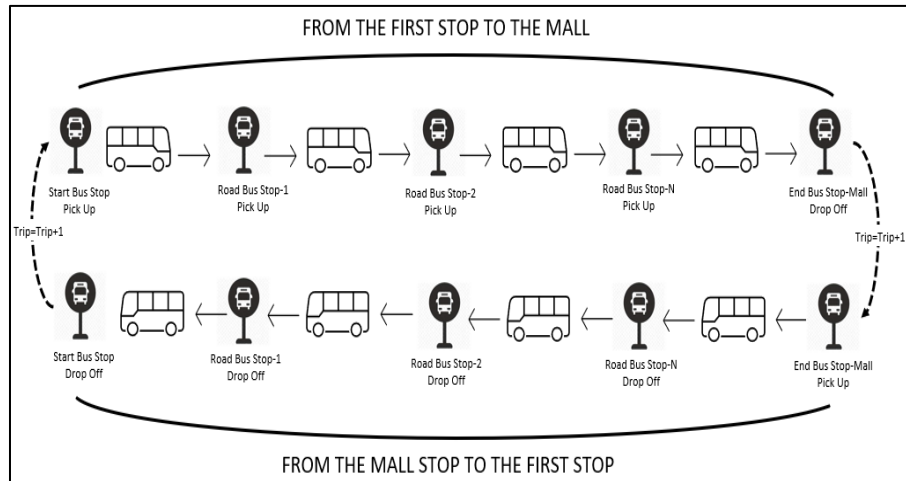


Figure 4.2 A visual example for movement logic of buses

Table 4.1 Example real-life schedule for route number 2

Route Number – 2	
From Start Bus Stop	From the Shopping Center
10:30	11:00
11:30	12:00
11:50	12:30
12:20	13:00
13:00	13:30
13:30	14:00
14:00	14:30
14:30	15:00
15:00	15:30
15:30	16:00
16:00	16:30
16:30	17:00
17:00	17:30
17:30	18:00
18:00	18:30
18:30	19:00
19:00	19:30
19:30	20:00
20:00	21:00
20:45	21:30

CHAPTER FIVE

SIMULATION MODELS OF A REAL BUS SERVICE SYSTEM

Simulation is one of the most important techniques for understanding the behaviour of a system and revealing its current state. Since the system subject to this thesis is a real-life system, it is not possible to do experiments on it and it is complex, simulation modelling method is preferred in order to see the current situation and understand the system dynamics. ARENA simulation software is used for simulation modelling. The simulation model is made with blocks and elements. The model created is a model that imitates the behaviour of the real system. Thus, the dynamics of the system became understandable. Thanks to this model, it is clearly seen how a change in system dynamics affects the overall performance of the system. The outline of the steps of the simulation modelling process is as follows.

1. Defining the point to be reached as a result of the model
2. Data requirement definition
3. Design for data collection and data collection process
4. Determining of simulation algorithm (Process flows)
5. Input analyse process
6. Pilot model building
7. Verification of the pilot model
8. Building of whole model
9. Verification of the whole model
10. Validation of the whole model

The modelling process begins by defining target of the model. It is extremely important to know what the established model will represent and what inferences should be drawn from the model. Otherwise, the established model will have difficulty in reflecting the real system, or even if it does, the results produced by the model will not be the desired results. For this reason, firstly, it is decided in which detail the model would be made and what would be desired to be seen as a result of the model. The information aimed to be reached as a result of the model is the average travel time. There is no information about the average travel times of the passengers in the real-life system. It is aimed to obtain this information with the simulation

model. Then, by changing the schedules of the buses in the system, the main purpose is to find the best average travel time.

After defining the purpose of the model to be established, the inputs of the model and which data should be used for these inputs must be defined. It is important to collect data that serves the defined purpose. In addition, the quality of the collected data is extremely important. Working with a quality data set means that the results of the simulation model represent reality. In this thesis, the number of passengers getting on and off the bus at the stops, and the time spent for boarding are basic data. In addition, real-life schedules, passenger capacities and routes of buses are among other data. An effective data collection method has been developed to work with a quality data set. A form has been prepared following the arrivals and departures of passengers at 134 different stops in the system. This prepared forum is filled daily by bus drivers. The information obtained from these forms is handled in MS Excel environment and made meaningful. In this developed form, the number of passengers getting on and getting down at each bus stop is kept. This form is filled in for each bus. The form developed for passenger tracking is given in figure 5.1.

Date							
		11:30	12:00	12:20	13:00	13:30	14:00
Bus Stop No	Bus Stop Name	Get On	Get Down	Get On	Get Down	Get On	Get Down
1	Evka5 Toki						
2	Evka5 Toki Yolu Girişi						
3	Köşe-2 Durağı						
4	Eski Son Durak						
5	Pazaryeri(cami)						
6	Tansaş Trafo Nokta						
7	Postahane						
8	Atatürk Mahallesi						
9	İlk Evler						
10	Park Manav İMKB Lisesi						
11	Okul						
12	Yeni Durak Ata						
13	Cami						
14	Santral						
15	Şaraphane						
16	Çiğli Kipa Son Durak						

Figure 5.1 Created sheet to track passenger movements in the system

In addition to this form, the length of time a bus spends on boarding and disembarking passengers when it approaches the stop is observed. The observations are made using the chronometer method. As a result of these

observations, an equation related to the number of passengers for the time spent getting on/off is created.

One of the basic inputs of a good simulation model is a good process flow. Simulation models represent a flow. Therefore, having a good understanding of the process flow in the real world will enable to establish a good model. The flow in the model is expected to be the same as the real-world flow. In order to make the best modelling, the process flows are drawn. Two different process flows are drawn from the eyes of the passenger and the bus driver. The two drawn process flows represent real-world behaviour. The simulation model is built according to the real-life process flows. The process flows drawn from the perspective of the passenger and the bus driver are given below in figure 5.2 and 5.3.

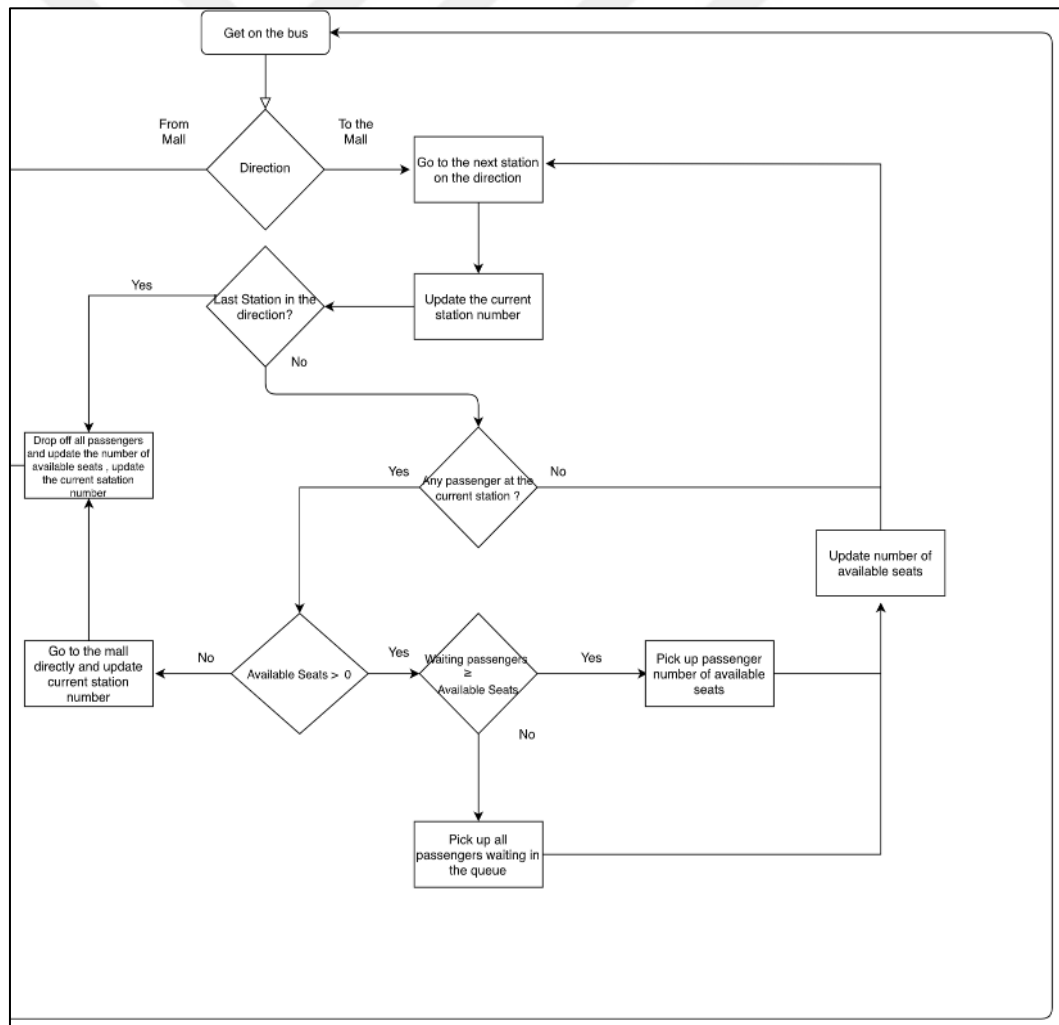


Figure 5.2 Process flowchart from the driver's perspective

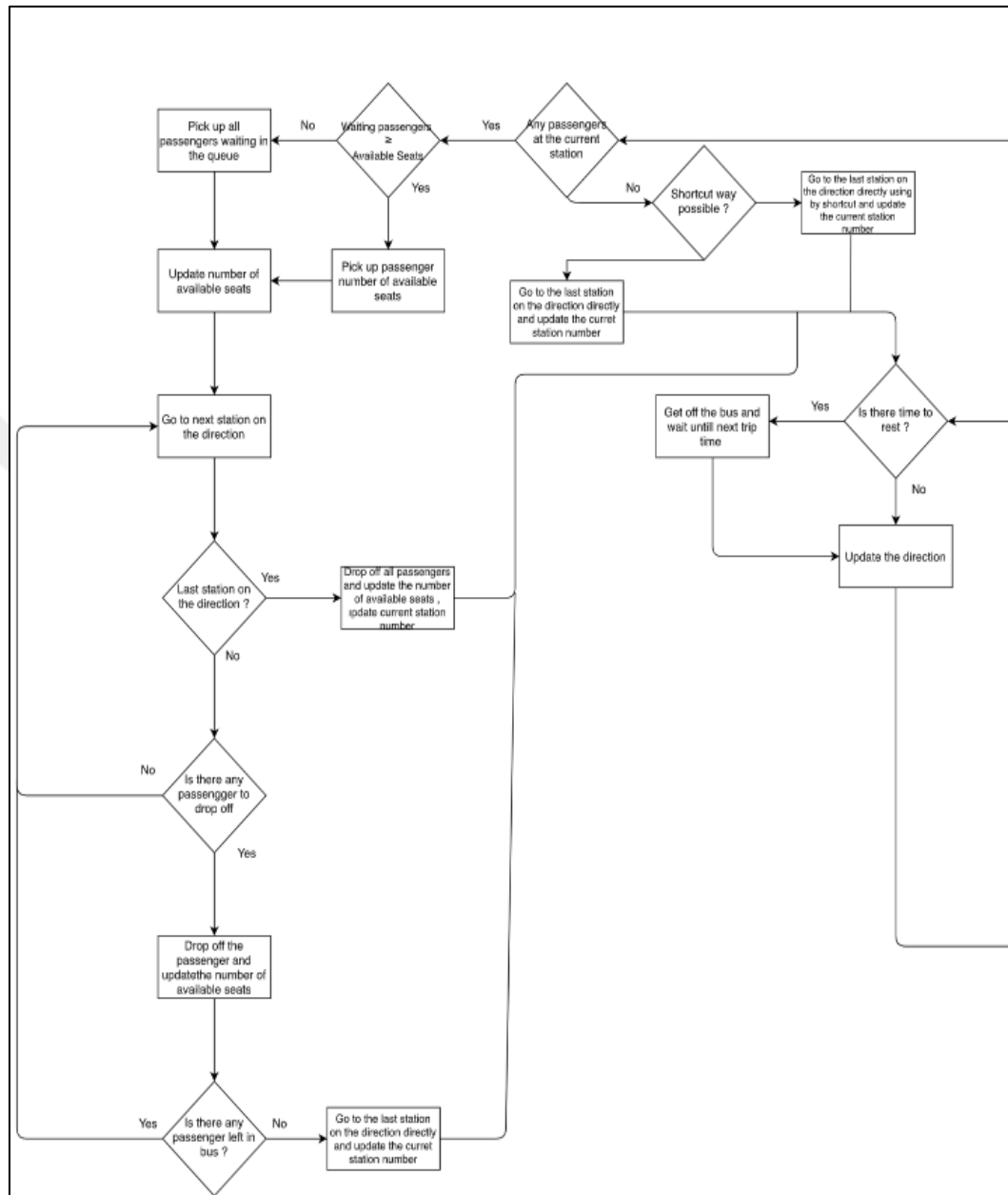


Figure 5.2 Continues

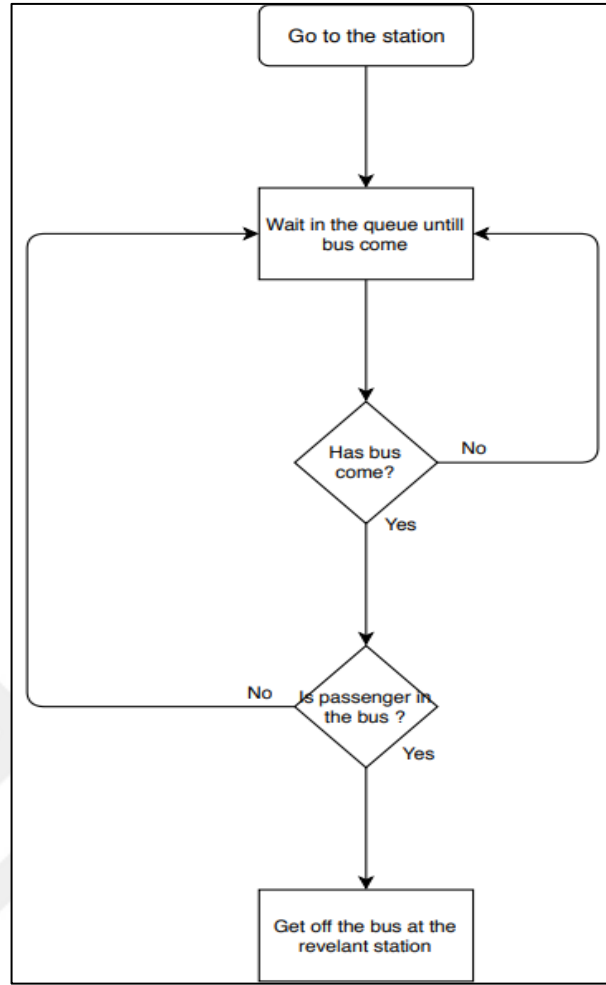


Figure 5.3 Process flowchart from the passenger's perspective

In the figure 5.4 below, there is an observation table prepared to create the equation representing the time of getting on/off the bus for the passengers in the table. The time of getting on/off the bus are observed by the stopwatch method.

Boarding/Unboarding Time (Sec)			Observations										Getting On		Getting Down	
Route	Plaque	Bus Number	1	2	3	4	5	6	7	8	9	10	Total	Average	Total	Average
Evka-5	35 KE 8135	1	15	27	16	18	58	21	34	13	27	25	134	26,8	120	24
Tersane	35 KE 8279	9	19	32	27	24	19	18	15	32	24	21	121	23,1	110	22
Bostanlı	35 AT 2447	14	27	31	15	16	13	16	25	16	23	34	102	21,6	114	22,8
Bahçelievler	35 KE 8384	17	16	22	18	44	37	25	30	22	13	19	137	24,6	109	21,8

Number of Passengers			Observations										Total	Average
Route	Plaque	Bus Number	1	2	3	4	5	6	7	8	9	10		
Evka-5	35 KE 8135	1	4	2	4	3	8	2	3	4	3	2	35	3,5
Tersane	35 KE 8279	9	3	4	3	4	2	3	2	3	4	3	31	3,1
Bostanlı	35 AT 2447	14	3	4	2	3	2	2	3	2	4	4	29	2,9
Bahçelievler	35 KE 8384	17	2	3	3	4	6	5	7	3	2	3	38	3,8

Figure 5.4 Observation form for time picking up and dropping off

All collected data are used in the simulation model. During the data collection phase, observations are made in the field and the available information is obtained

from the experts working in the transportation company. While the arrival/boarding times of the passengers and the number of passengers collected from the stops are obtained by the observation, information such as the number of buses, routes, bus capacities are obtained from the transportation company.

After the raw data are collected, they are handled with MS Excel. The processed data has been made meaningful in the Input Analyzer Module of the ARENA simulation software. Six different statistical distributions are obtained for each stop. In the model, passenger arrivals vary significantly on weekdays and weekends. In addition, there are three different time slots during the day: morning, noon and evening. Passenger demands vary significantly between these time periods. For this reason, six different statistical distributions are obtained for the passenger demands at each bus stop in the system. One of the most important points when obtaining statistical distribution is p significance value. The statistical distributions found represent the 95% confidence level. A detailed list of statistical distributions is given in the appendices. Sample input analysis output is given in figure 5.5 below.

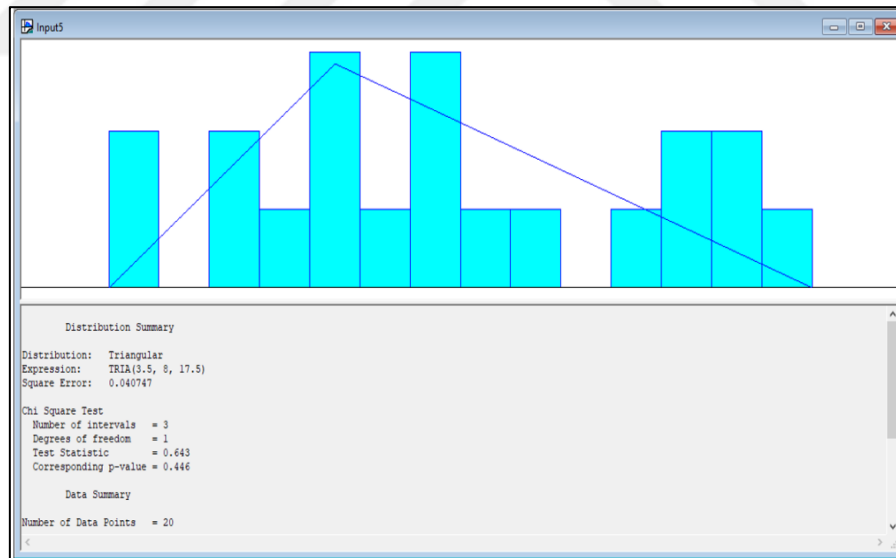


Figure 5.5 An example of input analyzer

Since the problem that is the subject of this thesis is a complex problem, two pilot models are made before the modelling phase. The first pilot model represents a single route, single bus system. The Bahçelievler route is chosen for the first pilot model. In the second pilot model, the Evka-2 route is added to the first pilot model. Thus,

two bus two route scenarios are studied in the second pilot model. Verification is made in order to determine the accuracy of the pilot models created. The verification process is done with the Trace element, which is one of the important elements of ARENA simulation software. With the Trace element, the suitability of each bus and each passenger in the simulation with the previously drawn flow has been determined. It has been observed that both pilot models act in accordance with the drawn process algorithms. This situation states that the logic of the simulation model represents real-life, and it is possible to proceed to the modelling phase of the whole system. Example output of the Trace element is given below figure 5.6.

SIMAN System Trace Beginning at Time: 0.0			
Seq#	Label	Block	System Status Change
Time: 0 Entity: 2			
1	0\$	CREATE	Arrival stream terminated Batch of 1 entities created
2	27\$	ASSIGN	Driver set to 1.0 Entity Station set to 22 Bahcelievler_Available_Seats set to 27.0 Direction set to 1.0 Bahcelievler_35KE8384_TravelNumber set to 1.0 Entity Sequence set to 1
3	28\$	REQUEST	Using velocity 5.0 Bahcelievler_35KE8384 request at station EnterStation Bahcelievler_35KE8384(1) to arrive at time 0.0 at block 28\$
4	24\$	TRANSPORT	Bahcelievler_35KE8384(1) transport to station Bahcelievler Bahcelievler_35KE8384(1) to arrive at time 10.0 at block 70\$ NC = 0.0
Time: 0 Entity: 4			
43	2\$	CREATE	Arrival stream terminated Batch of 1 entities created
44	30\$	ASSIGN	TimeSlot set to -3.0 Week_Days set to 0.0 Min_Velocity set to 500.0 Max_Velocity set to 600.0 Min_Velocity_Enterance set to 400.0 Max_Velocity_Enterance set to 600.0

Figure 5.6 Output of the Trace element

The verification of pilot models states that the simulation model created works correctly according to the logic of the real-life. Two different models have been created, emulating the real-life system, and covering all buses and stops. These are weekday and weekend models. While modelling in ARENA simulation software,

blocks and elements are used. The assumptions of the model representing the real-life system are as follows.

- Passengers start arriving to a stop half an hour before the arrival time of the first bus passing through the stop.
- Passenger's finish arriving at a stop at the last departure time of the last bus passing from that stop.
- The first passenger creation in simulation model start at the last stop is the same as the departure time of the bus on the route from the first stop.
- The first passenger creation in simulation model end at the last stop is the same as the departure time of the bus on the route from the last stop.
- Bus drivers get on the bus one minute before the departure time.
- Passengers in the system cannot leave the system without receiving service.
- Buses operating in the system do not delay/leave the system due to breakdowns.

Models that reflect the flow in the real-life system and contain these assumptions have been created.

The submodel feature of ARENA simulation software is used in the modelling process. Using submodels provides a more flexible model structure. First of all, the definitions of the buses, the current schedule and the current routes in the real-life system are made. In this study, the distances between the stops represent the actual distances. After all the definitions are made in the experimental frame, the model is built with the help of blocks. An example image from the main screen of the model is given in figure 5.7 and figure 5.8 below. Also experimental frame of the model given in figure 5.9 below.

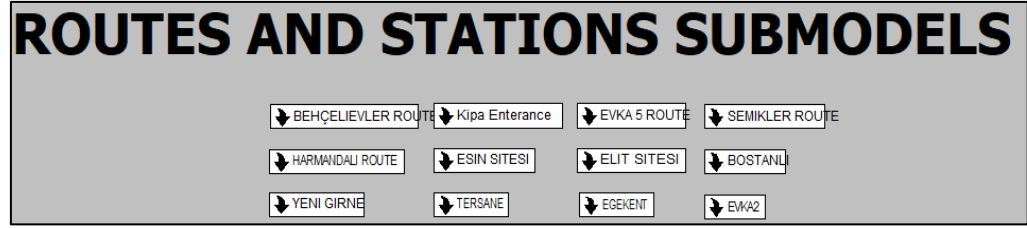


Figure 5.7 Submodels of the model



Figure 5.8 Submodels of the model

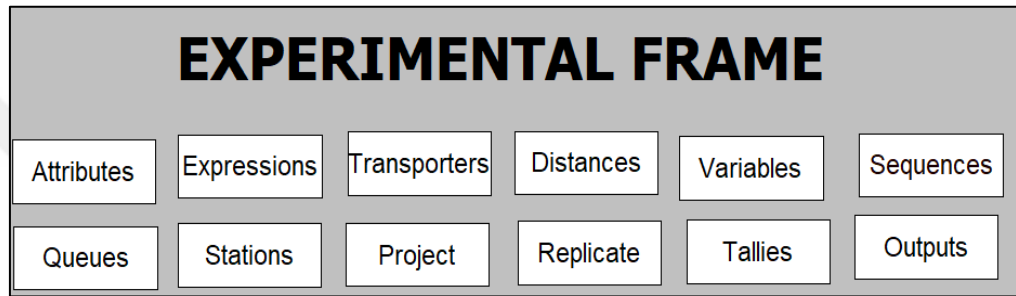


Figure 5.9 Experimental frame of the model

In the model, the flows of buses and passengers in the system are modelled. As a result of the built models, the average time spent by the passengers in the system is determined. The flow and critical points of the model are explained in detail in the following headings.

- ***Passenger arrivals***

In order to model the flow of passengers in the system, it is necessary to calculate the frequency of arrival of the passengers to the station. The data collected during the data collection phase are analysed and the statistical distributions of passenger arrivals at 134 different stops are found with the help of ARENA Input Analyzer. Statistical distributions representing the passengers' inter-arrival times are given in the appendices. Passenger arrivals in the system are made according to the determined statistical distributions. After a passenger arrives at the system, they go to the bus stop and wait for service.

Queuing logic at stops is designated as FIFO (first-come-first-served). Passengers wait at the bus stop until their bus arrives. In the simulation model, there must be enough seats on the bus to pick up a passenger from the stop. Otherwise, the passenger must wait for the next bus to get service. Passengers in the system wait at the bus stop until they receive service. In the real system, the arrival of passengers is not eternal. In the simulation model, a time is determined to stop the passenger arrivals. When the relevant time comes, the value of the variable called *stopcreatingentities* is reset, thus preventing passenger arrivals. Passengers who have entered the system after a certain period of time but cannot receive service due to the schedules of the buses in the system are directly sent out of the system with the help of the dispose block. The simulation model representing the passenger arrival is given in figure 5.10 below. Also, an example of create block is given in figure 5.11.

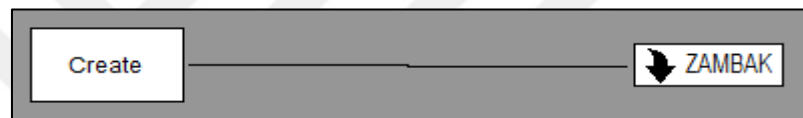


Figure 5.10 Passenger arrival model

Figure 5.11 Example create block for passenger arrivals

The formula written on the Interval tab in the Create block represents the passenger arrivals at the relevant bus stop. With the ED expression, it is provided to refer to the expressions element. All statistical distributions of the stops in the system are kept in the Expressions element. *Stopcreatingentities2* refers to a variable. The

initial value of this variable is set to be a very large number. The value of this variable is reset at a certain time. Thus, passengers are prevented from entering the system after a certain period of time. Logic of changing value of variables by time given in figure 5.12 below.

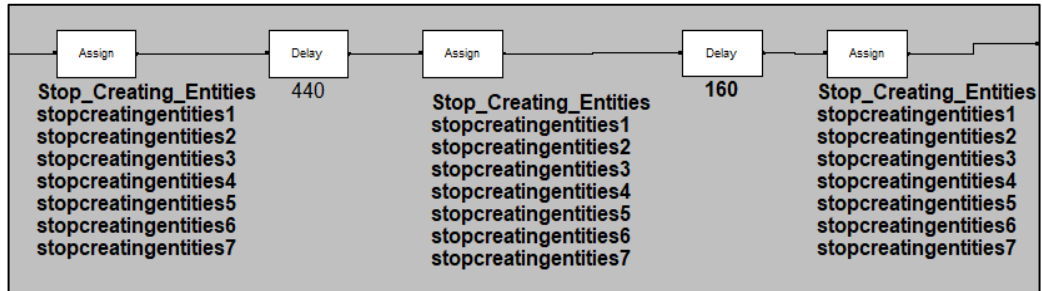


Figure 5.12 Logic of value changing for variables

The time slots are to change as the simulation time progresses. After time control entity directed to certain blocks within the simulation according to the entity type attribute. If the entity in the system is a passenger, it is sent directly to the queue. Logic of passenger movement model is given below in figure 5.13.

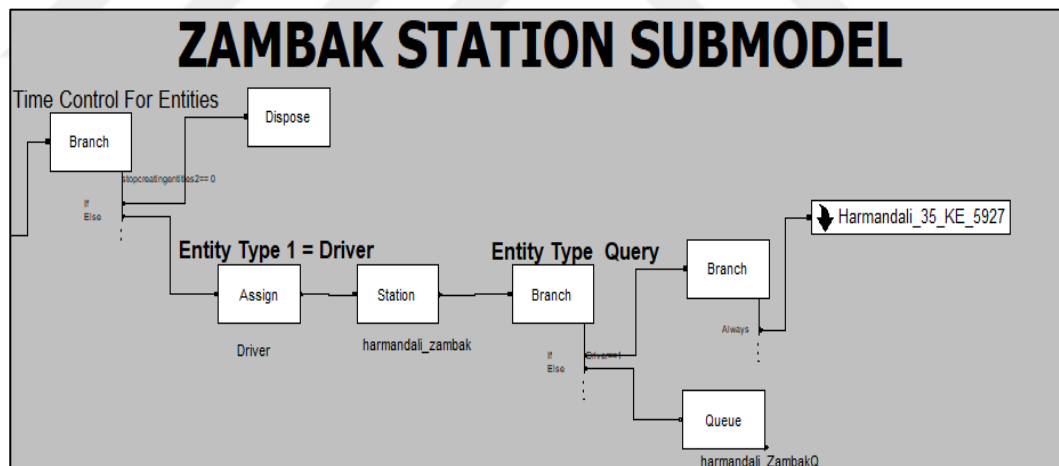


Figure 5.13 Passenger movement model

- **Creating bus drivers**

One of the entity types in the system is the drivers. Like passengers, drivers are created in create blocks and sent to the system. The drivers sent to the system keep their own buses with the help of the request block and drive the bus according to the

predetermined schedule and route. Images of the model part of the driver creation logic are given below. Creating driver model is given in figure 5.14 below.

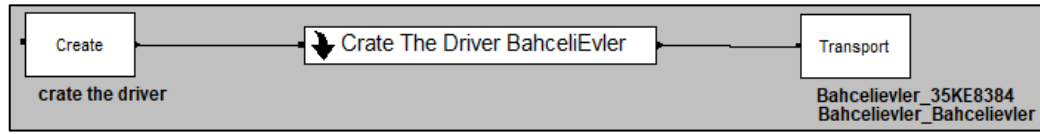


Figure 5.14 Creating driver model

Created drivers go into submodel then goes to Transport block. The Transport block ensures the bus to go on the specified route. Predetermined routes are kept inside the Sequences element. After a driver entity enters the submodel *direction*, *travel number*, *NS* attributes are assigned. The direction of the bus is important for the flow of the simulation model. The *NS* attribute is also extremely important. The *NS* attribute is a special expression of the ARENA and specifies which route a bus will take. The *NS* expression refers to the sequences element. On the other hand, the travel number of the bus is recorded with the variable travel number. Activating the bus by driver model given below in figure 5.15.

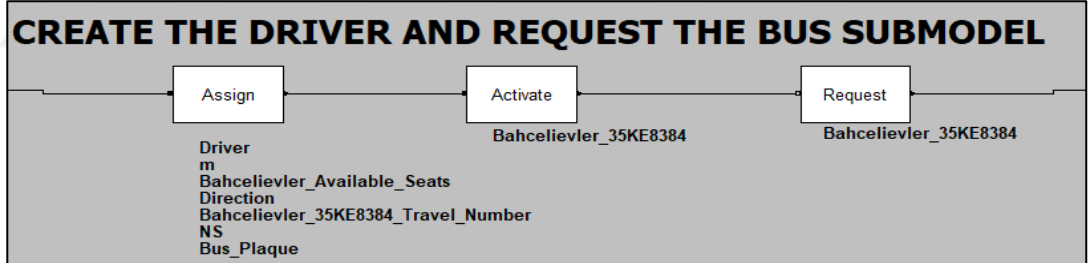


Figure 5.15 Activating the bus model

- **Transport between stations**

During the simulation run, buses follow a predetermined route and move between stops. Transports between stops are made with the help of the transport block. When a bus arrives at a stop, the current stop number is updated and departs to move to the next stop on the route. Example is given in figure 5.16.

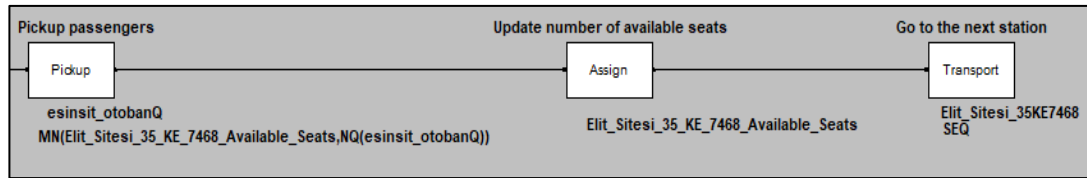


Figure 5.16 Transport between stations model

When the bus comes to a stop, first of all, the direction variable is checked in the simulation model. Then, the passenger drop-off or passenger pick-up processes are completed. After all the operations are completed, it moves in accordance with the bus route with the help of the transport block. Also, the value of the variable travel number is incremented by 1 each time the bus completes each trip. The image of an example transport block is given in figure 5.17 below.

Figure 5.17 Example of a transport block

SEQ expression is a special expression of ARENA. It refers to Sequences element. Also velocity is a key component of transporting between stops for buses. Velocities are changing according to the time slot. A special formula written in the Velocity tab in the Transport block ensures changing velocities according to the time slot during the simulation run. When buses arrive at the first stop on their route or the last stop on their route, they check the simulation time. The current simulation time is compared with the schedule of the bus. If the bus has lagged behind according to its predetermined schedule, it will continue on to the next trip without a break. However, in the opposite case, it waits at the relevant stop until the time in the

predetermined schedule arrives and does not pick up passengers. An example is given below in figure 5.18.

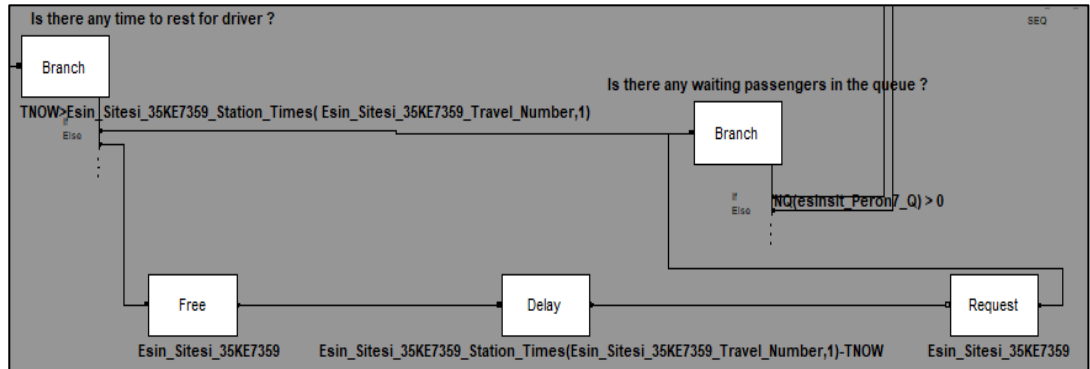


Figure 5.18 Example of bus movement

With the help of the branch block, the *TNOW* simulation time and the schedule time are compared. In case the bus lags behind the schedule, it must pick up the passengers and continue. In the opposite case, the bus is released with the Free block. The delay block allows the driver to wait until the next trip time. When the next trip time comes, the driver keeps the bus again with the help of the Request block and returns to its routine operations. Predetermined schedules kept in Variables element. These schedules are schedules of the real system. Also the distances between bus stops are real distances and they kept in Distances element.

- **Passenger pick-up**

One of the important components of the simulation model is picking up passengers. Three different controls are provided in the simulation model when a bus arrives at the stop to pick up passengers. The first control is whether there are available seats on the bus. If there are no available seats on the bus, the bus will go directly to the last stop on its route. However, if there are available seats on the bus, the second control process begins. In the second control process, it is checked whether there are passengers at the bus stop. If the number of passengers waiting at the stop is greater than zero, the third control process begins. If the number of passengers waiting at the stop is less than zero, the bus moves to the next stop within the route. In the third control process, the number of passengers waiting at the bus stop and the number of empty seats on the bus are compared. The bus picks up the

minimum number of passengers waiting at the stop and the number of empty seats and moves to the next stop on its route. Every time the bus picks up passengers from the bus stop, the available seats variable in the bus is updated. The initial value of the variable seats variable is set to 27. This value is determined according to the capacity of the buses in the real system. An example figure 5.19 is given below which is related with first and second control process.

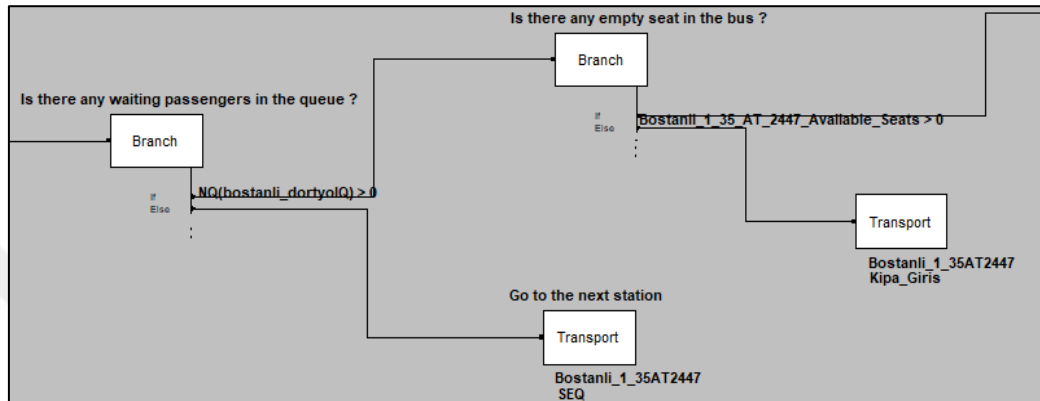


Figure 5.19 First and second control logic for buses

- **Passenger drop-off**

Another important component in the simulation model is the drop off logic. When the bus arrives at the stop to drop passengers, it checks whether there are passengers to drop off at the relevant stop. There is a distance ID for each station. When the passengers arrive at the stop, a distance ID is assigned to the passengers according to the stop they will drop off at. When the bus arrives at the stop, the distance ID value of the relevant stop is compared with the distance ID values of the passengers inside the bus. This comparison is made with the help of the search block. If there is a passenger who meets the search criteria, a value is automatically assigned to the J variable, which is the special variable of the ARENA. If the value of J is greater than zero, it means that there is a passenger to get off at the relevant stop. If the J value is greater than zero, the passenger or passengers drop off the bus and then continue to the other stop within the bus route. If the value of J is equal to zero, it means that there is no passenger to drop off at the stop. In this case, the bus moves to the other stop within its route without any break. Each passenger arriving at the mall stop is

assigned a distance ID with the discrete function. Distance ID's are given in the appendices. In addition, the representation of the relevant logic in the model is given in figure 5.20 below.

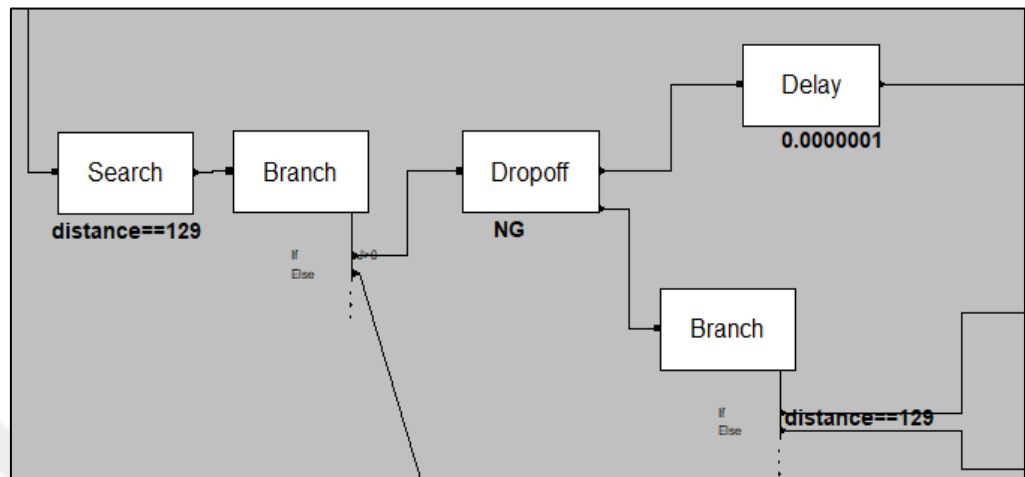


Figure 5.20 Drop-off logic for buses

- **Recording average travelling time**

Average travel time is a KPI that cannot be measured in the real system. The main purpose of the built model is to measure the average travel time of the passengers and to understand the dynamics affecting this average travel time. The average travel time in the simulation model is measured with the help of the Tallies element and the Tally block. An attribute called *TimeIn* is defined for each passenger who logs into the system. The *TimeIn* attribute is an attribute whose value changes as the simulation time increases after the passenger logs into the system. While the passenger leaves the system, the average time spent by the passenger in the system is calculated with the help of the Tally block. The *INT* expression, which is a special expression of ARENA written in the Tally block, measures the time interval between the passenger entered the system and exit. An example image of Tally block given in figure 5.21 below.

The screenshot shows a 'Tally Block' dialog box. It has a title bar with a question mark and a close button. The main area contains several input fields: 'Label' (empty), 'Mark Attribute' (dropdown menu), 'Next Label' (empty), 'Tally ID' (dropdown menu showing 'Passengers_DroppedoffTimein'), 'Value' (text box containing 'INT(Timein)'), and 'Number of Observations' (text box containing '1'). There is also a 'Comments' section with a text area. At the bottom are 'OK', 'Cancel', and 'Help' buttons.

Figure 5.21 An example of tally block

After the weekday and weekend models are built, the model needs to be verified. The verification process checks whether the model fits the real-life flow. The Trace element is used for the verification process. With the Trace element, the flows of all buses and passengers in the system are observed in the model. Thus, it is ensured whether the system works in accordance with the real-life system or not.

After the controls with the Trace element, the model has been verified and it has been observed that the model flow is suitable for the real-life flow. However, the validation process must be completed to prove that the results from the model are significant and represent the real system. The validation process checks the compatibility of the results produced by the model with the real-life system results. At this point, the average number of passengers transported during the day is considered as the main validation indicator. The number of passengers transported in the real-life system is known. For this reason, the average number of passengers transported is considered as the validation indicator of the simulation model. The Output Analyzer module of ARENA simulation software is used for the validation process. Weekday and weekend models are run for 10 replications. The number of passengers transported in both models is recorded in a file with a *.dat* extension. In the Output Analyzer module, a 95% confidence interval has been built for the data in this file. The average number of passengers transported in a day in the real-life system is within this confidence interval. Thus, the validation of the model is completed. It

can be said that the simulation model created based on the results of the validation process reflects the results of the real-life system. An image of the confidence interval in the Output Analyzer module is given below in figure 5.22.

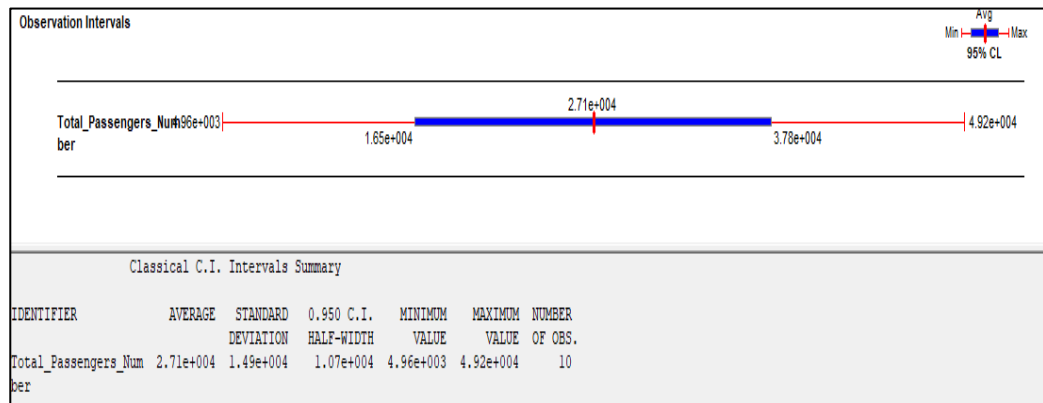


Figure 5.22 Output analyzer result for weekday model

CHAPTER SIX

SIMULATION OPTIMIZATION APPROACH FOR SOLUTION TO A REAL BUS SERVICE SYSTEM

Optimization is the process of finding the best value for a problem under certain constraints. In this thesis, the schedules of buses are tried to be optimized with metaheuristic methods. A feasible schedule for each bus in the system are created with metaheuristic methods. The created feasible optimal/near optimal schedules are the schedules that minimize the average travel time of the passengers in the system. The average travel time includes the passenger's waiting time at the stop and the time spent on the bus.

There are four different algorithms used in the optimization process. These algorithms are GA, ACO, PSO and ABC. Algorithms are coded in VBA environment and integrated into ARENA simulation software. Thus, the integrated operation of the simulation model and optimization codes is ensured. The scheduling problem discussed in this thesis is an NP-hard problem type. In this problem, which deals with the real-life system, there are nine different schedules that 20 different buses can use. From this point of view, it can be said that the size of the solution space is very large, and the problem has 9^{20} different candidate solutions. Therefore, it can be said that the problem addressed in this thesis is an NP-hard problem. Since the problem is an NP-hard problem, artificial intelligence-based metaheuristic methods are used instead of exact methods, and metaheuristic algorithms are integrated into ARENA simulation software to reach the result faster.

In the real-life system, each bus in the system has its own schedule to be obliged to follow. These schedules are given in the appendices. Current schedules include the starting time of the bus to each trip. The trip starting times of the buses are moved forward and backward for 20 minutes at 5 minute intervals. This process has been done for all buses. In the optimization process, the most suitable and feasible combination of schedules has been tried to be found. The details of the metaheuristic algorithms used in the optimization process are explained in the following headings.

6.1 Genetic Algorithm

GA is a method frequently used in the literature to solve complex engineering problems and real-life cases. GA is inspired by human biology and the structure of human DNA (Gendreau, & Potvin, 2019). One of the solution methods for the bus scheduling problem, which is the subject of this thesis, is the GA.

6.1.1 Encoding Process for Genetic Algorithm

The first step to solve the bus scheduling problem using GA is to express the real-life problem in the language of the GA. This process is called the encoding process. The real-life system has 20 different buses and 20 different schedules. In this problem, 20 different schedules that already exist are stretched forward and backward for 20 minutes at 5-minute intervals. The schedules of all buses are changed at 5-minute intervals and the best combination is tried to be found. Each combination represents a candidate solution. In the GA, each solution is called a chromosome. The schedules used by buses in real-life correspond to the chromosomes in the GA. In other words, the chromosomes represent the schedules used by 20 different buses. A gene can take nine different values. The figure 6.1 below shows the real-life problem from the GA point of view.

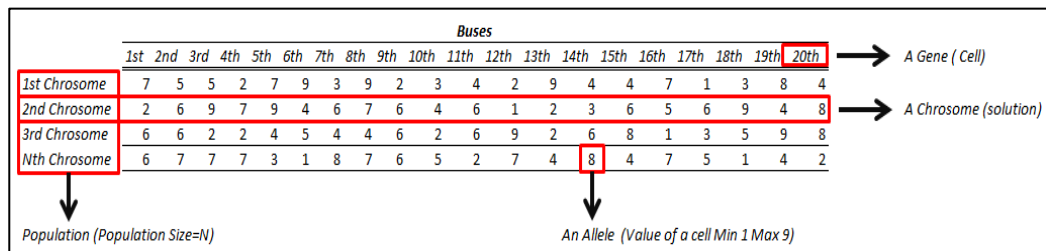


Figure 6.1 Representation of problem with GA view

Each bus in a real-life problem shown by a gen in a chromosome. Genes combine to form chromosomes. In fact, a chromosome is a combination of schedules used by buses. Each chromosome represents a candidate solution in the solution space. Chromosomes come together to form the population. The values of the genes depicts the schedules used by the buses. There is five minutes difference between schedule number one and schedule number two. The difference between schedule number one and number three is 10 minutes. Schedule 5 shows the current schedule of the bus.

6.1.2 Flowchart of Genetic Algorithm for a Real-Life Bus Scheduling Problem

The GA consists of many stages. Each step in the GA is critical, as the failure to implement any of these steps will invert the used algorithm in a different direction. The process that starts with the definition of the parameter continues with the creation of the population, the selection of the chromosomes of the parents, the production of children and the control of the criteria. The general flow of the GA used for the bus scheduling problem is given in the diagram below figure 6.2.

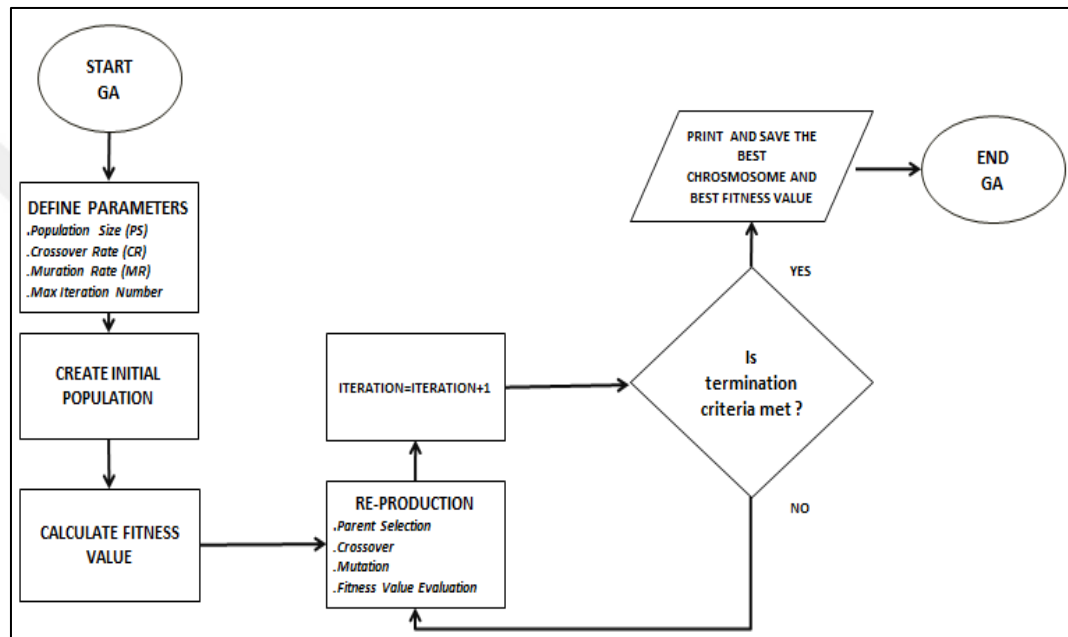


Figure 6.2 Flowchart of GA

6.1.3 Procedures of Genetic Algorithm for a Real-Life Bus Scheduling Problem

Each step in the GA has its own procedure. The GA procedures and pseudo codes used in solving the problem that is the subject of this thesis are explained in detail below table 6.1.

Table 6.1 Pseudo codes of GA

BEGIN GENETIC ALGORITHM SET PARAMETERS PS ,CR, MR, MaxIter <i>For IterationCounter=0 To MaxIter-1</i> GENERATE INITIAL POPULATION RANDOMLY <i>From PS=0 to PS-1</i> ARRANGE SIMULATION MODEL Set schedules according to the population RUN ARRANGED SIMULATION MODEL CATCH FITNESS VALUES EVALUATE FITNESS VALUES Rank the chromosomes from the best towards the worst SELECT PARENTS FOR NEW GENERATION Select the parents using greedy selection method CROSSOVER <i>Ncrs=CR*PS, Determine number of chromosome selected for crossover</i> <i>For var=0 to Ncrs-1</i> Determine crossover point Create children using single point crossover <i>Next var</i> Add children to the population MUTATION <i>Nmtt=MR*PS, Determine number of chromosome selected for mutation</i> Specify a gene to mutate using random numbers then mutate RE-ARRANGE SIMULATION MODE Set schedules according to the population CATCH FITNESS VALUES EVALUATE FITNESS VALUES Find the best chromosome and the best fitness value and record it for current iteration Find the best chromosome and the best fitness value and record it for all iterations UPDATE ITERATION COUNTER <i>Iterationcounter=Iterationcounter+1</i> CHECK TERMINATION COUNTER <i>Next Iterationcounter</i> REPORT THE RESULT AND END GENETIC ALGORITHM

Codes are written in ARENA simulation software's VBA environment. Thus, the integrated run of the optimization codes and the simulation model is ensured. Written optimization codes are integrated with both the weekday model and the weekend model. Procedures of GA are explained below.

- ***Parameter definition procedure***

As mentioned before, *CR*, *MR* and *PS* parameters have significant effects on the success of the algorithm. Therefore, the parameter setting procedure is extremely important. However, when looked to the literature, it is not possible to say that a certain combination of parameters is successful. The most successful *CR*, *MR* and *PS* parameter combinations vary according to the dynamics of the studied problem. Therefore, 27 different combinations are used in the optimization process with the GA. The used combinations are given in table 6.2 below.



Table 6.2 Parameter combinations for GA

	PS	CR	MR
<i>Combination-1</i>	20	0.4	0.2
<i>Combination-2</i>	20	0.4	0.4
<i>Combination-3</i>	20	0.4	0.6
<i>Combination-4</i>	20	0.5	0.2
<i>Combination-5</i>	20	0.5	0.4
<i>Combination-6</i>	20	0.5	0.6
<i>Combination-7</i>	20	0.8	0.2
<i>Combination-8</i>	20	0.8	0.4
<i>Combination-9</i>	20	0.8	0.6
<i>Combination-10</i>	40	0.4	0.2
<i>Combination-11</i>	40	0.4	0.4
<i>Combination-12</i>	40	0.4	0.6
<i>Combination-13</i>	40	0.5	0.2
<i>Combination-14</i>	40	0.5	0.4
<i>Combination-15</i>	40	0.5	0.6
<i>Combination-16</i>	40	0.8	0.2
<i>Combination-17</i>	40	0.8	0.4
<i>Combination-18</i>	40	0.8	0.6
<i>Combination-19</i>	100	0.4	0.2
<i>Combination-20</i>	100	0.4	0.4
<i>Combination-21</i>	100	0.4	0.6
<i>Combination-22</i>	100	0.5	0.2
<i>Combination-23</i>	100	0.5	0.4
<i>Combination-24</i>	100	0.5	0.6
<i>Combination-25</i>	100	0.8	0.2
<i>Combination-26</i>	100	0.8	0.4
<i>Combination-27</i>	100	0.8	0.6

- ***Initial population creation procedure***

The initial population created at the beginning of the algorithm is created completely randomly. The number of solutions in the population, that is, the number of chromosomes, is equal to the *PS* parameter determined at the beginning of the algorithm. The important point at this stage is the values of the genes in the chromosomes. These values are called alleles. The lower and upper limits of allele values should be determined. For this problem, the lower allele has a value of 1 while

the upper allele has a value of 9. Values consisting of random integer numbers within the lower and upper limits are created for each gene. Genes come together to form chromosomes, and chromosomes come together to form a population.

- ***Fitness value evaluation procedure***

After the initial population has formed, the simulation model has run. Thus, fitness values are caught. After fitness values are caught, population is ranked by fitness values from the best to the worst. Thus, the best chromosome is determined in the population. The best fitness value is recorded in the algorithm. To catch fitness values simulation model must be run. In the VBA environment of ARENA simulation software there are some special commands. Once simulation model starts to run, codes arrange simulation model and change the value of genes. Thus, the algorithm generates the new schedule. After simulation model arrangement, simulation model runs and generates the fitness values. Fitness value is the average travelling time.

- ***Parent selection procedure***

Parent selection process directly affects the next generations and the success of algorithm. Algorithm specifies the parents to create children. One of parent called mother and the other one is called father chromosome. There can be more than one couple but number of couples must be even number. In the literature there are many parent selection methods. In this thesis, *greedy selection* method is used for parent selection process. This method chooses the pre-defined number of chromosomes as parent from the ranked population. Thus, the best chromosomes are selected as parents and the probability of being more successful for new generations increased. The step in which learning takes place in the algorithm is the parent selection procedure. With the greedy selection technique, the best chromosomes are selected as parents and the characteristics of the best solution are transferred to the next generations.

- ***Crossover procedure***

Parents are selected to generate new children, obtain new solutions and fitness values. Crossover is one of the most important operators of GA. With this operator

new chromosomes are generated. Half of selected parents represents mother, and the other half represents father. New child forms from genes of mother and father. In this thesis *single point* crossover method is used to generate children. The algorithm specifies a crossover point using random numbers. Specified random numbers has upper and lower bound. The lower bound is two and upper bound is the number of genes-1. The genes of the maternal chromosome before the determined crossover point and the genes of the paternal chromosome after the determined crossover point come together to form a child. With the crossover operator, the algorithm searches for different points of the solution space randomly. Figure 6.3 below explains the crossover procedure.

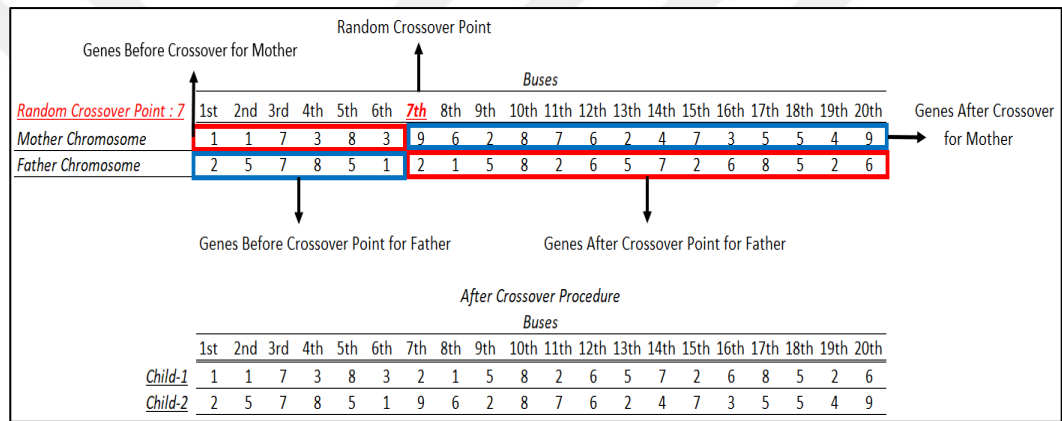


Figure 6.3 An example for crossover procedure

- **Mutation procedure**

Mutation operator is also one of the important operator of GA. Thanks to this operator the algorithm searches different points of solution space and discovers new areas in solution space. In this thesis pre-define number of genes has mutated. In mutation procedure, a gene or genes of a chromosome chosen for mutation. Mutation operator changes the value of a gene randomly. Thus, diversity is ensured in the population. Mutated chromosomes added to the population and replaced with the old one. Figure 6.4 below explains the crossover procedure.

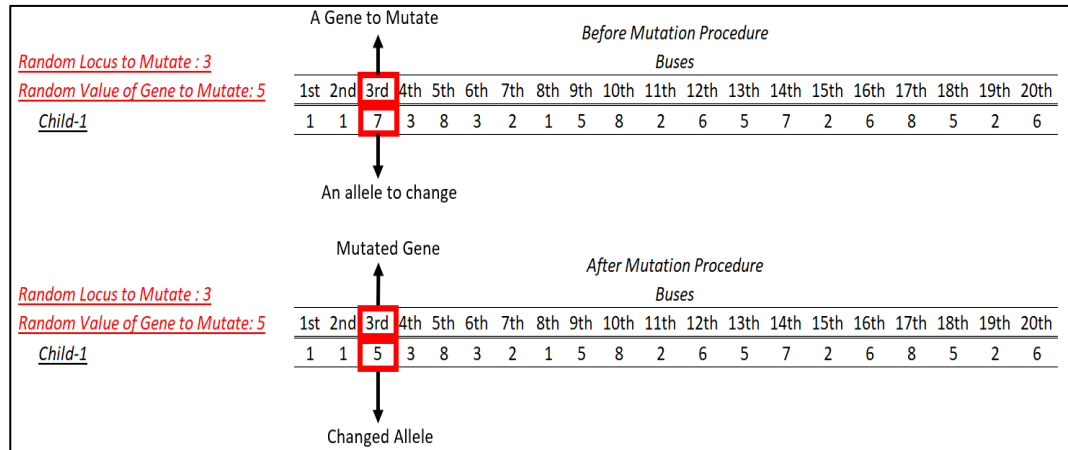


Figure 6.4 An example for mutation procedure

6.1.4 Results of Genetic Algorithm

The GA is run for 27 different parameter combinations. Each GA iteration includes 20 simulation replications. The results obtained by running the GA are summarized in the table 6.3 and 6.4 below.

Table 6.3 Results of GA for the weekdays model

Weekdays Model				The Best Solution	Fitness Value
	PS	CR	MR		
Current	-	-	-	5-5-5-5-5-5-5-5-5-5-5-5-5-5-5-5-5-5-5-5-5-5	52.614
Combination-1	20	0.4	0.2	1-1-7-3-2-8-7-3-3-6-5-6-7-6-9-1-4-5-2-4	47.938
Combination-2	20	0.4	0.4	9-5-6-1-1-4-9-3-2-5-5-3-9-2-9-3-5-1-2-1	47.962
Combination-3	20	0.4	0.6	4-6-6-6-9-7-1-3-2-4-3-2-1-3-4-4-6-4-8-8	47.774
Combination-4	20	0.5	0.2	2-4-4-5-3-2-6-3-7-5-5-3-6-8-5-6-8-2-7-4	46.829
Combination-5	20	0.5	0.4	3-6-2-8-3-1-2-6-6-7-5-6-9-9-4-1-1-6-1-6	46.933
Combination-6	20	0.5	0.6	2-2-7-3-8-2-3-6-7-1-2-2-6-6-9-4-1-8-8-1	47.902
Combination-7	20	0.8	0.2	8-3-2-4-5-9-5-1-9-1-1-7-9-9-9-1-7-3-9-9	47.832
Combination-8	20	0.8	0.4	3-8-4-1-7-2-1-9-9-7-2-7-5-6-3-5-5-5-9-9	47.850
Combination-9	20	0.8	0.6	5-3-7-3-9-1-6-6-1-4-9-5-7-9-5-7-6-3-9-9	47.970
Combination-10	40	0.4	0.2	3-2-1-4-1-6-5-4-2-1-8-2-6-2-6-7-3-6-1-4	47.735
Combination-11	40	0.4	0.4	2-3-5-3-9-2-7-8-5-1-8-2-6-8-2-4-3-6-4-4	47.724
Combination-12	40	0.4	0.6	5-2-6-1-1-8-8-6-2-3-1-1-1-3-4-8-2-6-7-5	47.720
Combination-13	40	0.5	0.2	8-5-3-3-4-5-4-5-3-4-6-4-6-9-3-1-4-5-9-9	47.620
Combination-14	40	0.5	0.4	4-8-8-6-4-6-6-6-1-8-1-2-3-4-4-8-1-9-3-1	47.615
Combination-15	40	0.5	0.6	3-6-8-8-8-1-2-5-5-3-4-8-1-8-3-2-6-1-6-5	47.505
Combination-16	40	0.8	0.2	5-5-4-4-4-3-1-5-5-2-2-7-1-5-1-1-9-2-3-6	46.539
Combination-17	40	0.8	0.4	4-6-3-6-4-1-8-6-6-7-8-2-3-5-4-9-5-4-4-4	46.650
Combination-18	40	0.8	0.6	5-1-4-2-5-3-2-9-8-3-1-5-5-7-5-7-4-6-3-4	46.819
Combination-19	100	0.4	0.2	2-4-1-7-1-4-3-9-9-2-2-5-9-3-5-8-5-9-6-9	46.654
Combination-20	100	0.4	0.4	4-4-2-1-7-3-1-5-5-6-8-2-4-3-2-4-4-7-8-4	46.664
Combination-21	100	0.4	0.6	4-4-2-1-7-3-1-5-5-6-8-2-4-3-2-4-4-7-8-4	46.664
Combination-22	100	0.5	0.2	4-2-1-6-9-2-4-8-6-9-8-1-2-5-8-1-4-8-5-4	46.454
Combination-23	100	0.5	0.4	5-5-4-4-4-3-1-5-5-2-2-7-1-5-1-1-9-2-3-6	46.539
Combination-24	100	0.5	0.6	3-5-8-2-3-4-7-6-5-2-3-4-8-7-1-4-8-7-8-6	46.530
Combination-25	100	0.8	0.2	3-9-8-6-2-8-2-5-3-1-3-6-4-5-1-7-3-8-1-1	46.239
Combination-26	100	0.8	0.4	8-5-5-1-6-2-7-1-3-8-5-3-8-7-5-8-7-8-6-8	45.812
Combination-27	100	0.8	0.6	3-5-1-7-6-4-1-8-6-5-4-1-6-5-3-5-3-1-7-5	47.434

Table 6.4 Results of GA for the weekend model

Weekend Model	PS	CR	MR	The Best Solution	Fitness Value
<i>Current</i>	-	-	-	5-5-5-5-5-5-5-5-5-5-5-5-5-5-5-5-5-5-5-5-5-5	78.856
<i>Combination-1</i>	20	0.4	0.2	6-3-5-2-4-1-1-7-8-4-4-8-2-3-5-7-7-9-2-8	69.380
<i>Combination-2</i>	20	0.4	0.4	2-7-7-8-8-6-3-5-6-1-2-9-3-8-6-2-8-2-3-1	69.775
<i>Combination-3</i>	20	0.4	0.6	3-3-4-6-6-9-7-3-4-2-3-3-8-8-4-5-8-6-6-8	69.351
<i>Combination-4</i>	20	0.5	0.2	8-5-3-3-4-5-4-5-3-4-6-4-6-9-3-1-4-5-9-9	69.758
<i>Combination-5</i>	20	0.5	0.4	2-2-6-9-1-1-4-7-8-5-2-4-4-9-5-6-8-7-3-1	69.658
<i>Combination-6</i>	20	0.5	0.6	8-6-5-6-5-3-9-6-3-1-2-1-4-9-4-7-8-8-2-2	69.905
<i>Combination-7</i>	20	0.8	0.2	2-7-7-8-8-6-3-5-6-1-2-9-3-8-6-2-8-2-3-1	69.775
<i>Combination-8</i>	20	0.8	0.4	2-7-7-8-8-6-3-5-6-1-2-9-3-8-6-2-8-2-3-1	69.775
<i>Combination-9</i>	20	0.8	0.6	8-7-6-5-4-9-1-5-2-6-2-1-7-3-2-8-2-4-8-1	69.853
<i>Combination-10</i>	40	0.4	0.2	5-3-3-1-5-7-3-3-2-2-3-2-6-4-3-8-4-9-4-9	69.433
<i>Combination-11</i>	40	0.4	0.4	7-5-4-8-5-1-1-4-3-4-1-2-5-5-7-5-6-5-1-5	69.539
<i>Combination-12</i>	40	0.4	0.6	9-3-5-8-2-4-3-7-3-1-7-5-3-4-2-4-3-2-9-5	69.527
<i>Combination-13</i>	40	0.5	0.2	7-4-2-8-2-7-4-2-2-1-7-7-2-3-1-7-9-1-6-4	69.508
<i>Combination-14</i>	40	0.5	0.4	4-6-5-4-1-1-3-1-4-3-3-1-4-3-2-5-5-4-3-6	69.641
<i>Combination-15</i>	40	0.5	0.6	7-5-6-3-3-7-1-7-8-7-1-4-8-8-4-9-8-1-9-4	69.706
<i>Combination-16</i>	40	0.8	0.2	2-7-2-1-3-6-4-7-5-3-2-4-4-4-9-4-4-9-3-3	69.613
<i>Combination-17</i>	40	0.8	0.4	8-4-7-2-1-5-1-3-6-8-6-3-5-4-4-3-4-8-4-7	68.526
<i>Combination-18</i>	40	0.8	0.6	3-7-5-6-8-4-7-9-7-7-9-1-3-2-7-3-3-7-8-7	69.166
<i>Combination-19</i>	100	0.4	0.2	4-3-3-5-6-1-1-4-1-6-3-3-1-8-2-3-6-9-2-4	68.740
<i>Combination-20</i>	100	0.4	0.4	8-4-8-2-1-6-1-5-3-2-3-3-2-8-1-2-2-4-8-5	68.965
<i>Combination-21</i>	100	0.4	0.6	4-3-6-7-7-8-4-8-7-6-9-2-3-2-3-3-5-8-6-9	69.429
<i>Combination-22</i>	100	0.5	0.2	2-5-4-9-4-2-9-6-7-9-9-5-7-9-5-3-4-1-2-2	69.409
<i>Combination-23</i>	100	0.5	0.4	1-8-1-9-6-1-4-9-3-7-9-3-3-3-3-4-7-4-7-2	69.399
<i>Combination-24</i>	100	0.5	0.6	8-4-7-2-1-5-1-3-6-8-6-3-5-4-4-3-4-8-4-7	68.526
<i>Combination-25</i>	100	0.8	0.2	3-1-1-5-1-6-1-3-2-8-5-6-8-8-4-6-6-8-5-5	67.146
<i>Combination-26</i>	100	0.8	0.4	4-7-6-6-4-4-4-5-7-7-9-9-7-9-3-3-3-3-9-6	67.780
<i>Combination-27</i>	100	0.8	0.6	4-1-3-4-9-4-4-7-7-6-5-2-5-5-5-5-9-3-7-1	68.468

GA run for 27 different parameter combination and provided 12.92% improvement for the weekday model and 14.84% for the weekend model. As it predicted, higher population size and crossover rate has provided better solutions than the others. The best parameter combination for weekend model is 100/0.8/0.2 while the best parameter combination for weekday model is 100/0.8/0.4.

6.2 Ant Colony Optimization

ACO is also a metaheuristic optimization algorithm used to solve complex problems. This algorithm is inspired by the behaviour of ants in nature. The algorithm is about ants reaching their food sources in the shortest time possible (Dorigo, Birattari & Stutzle, 2006). Although the ACO algorithm is generally used in traveling salesman problems, it is also an algorithm that can be adapted to scheduling

problems. One of the solution methods of the bus scheduling problem, which is the subject of this thesis, is the ACO algorithm.

6.2.1 Encoding Process for Ant Colony Optimization

The first step to solve the bus scheduling problem using ACO is to express the real-life problem in the language of the ACO. This process is called the encoding process. The real-life system has 20 different buses and 20 different schedules. In this problem, 20 different schedules that already exist are stretched forward and backward for 20 minutes at 5-minute intervals. The schedules of all buses are changed at 5-minute intervals and the best combination is tried to be found. Each combination represents a candidate solution. Ants in the ACO algorithm represent a candidate solution. Ants are initially assigned to random nodes. Each node represents a bus and the value of each node represents the schedule used by the bus. The ants in the algorithm come together to form the population. Each ant in the algorithm goes to as many nodes as the number of buses. When ants choose the nodes they will go to, they consider the probability of choosing the nodes and the amount of pheromone between the nodes. Ants in nature leave a chemical liquid that is directly proportional to the quality of the solution reached and inversely proportional to the length of the route. This chemical liquid is called a pheromone. Other ants in the colony determine the path they will go according to the density of the amount of pheromone released. The steps of the algorithm are repeated for a predetermined number of iterations. When the stopping criterion of the algorithm is met, the algorithm stops and reports the results.

6.2.2 Flowchart of Ant Colony Optimization for a Real-Life Bus Scheduling Problem

The ACO algorithm consists of many steps. Each of these steps plays an important role in the success of the algorithm. The algorithm first starts with the definition of parameters. The population is created according to the defined parameters, the fitness values are checked, and the ants in the system go to different nodes and create different solutions. The flowchart of the algorithm used is given below in figure 6.5.

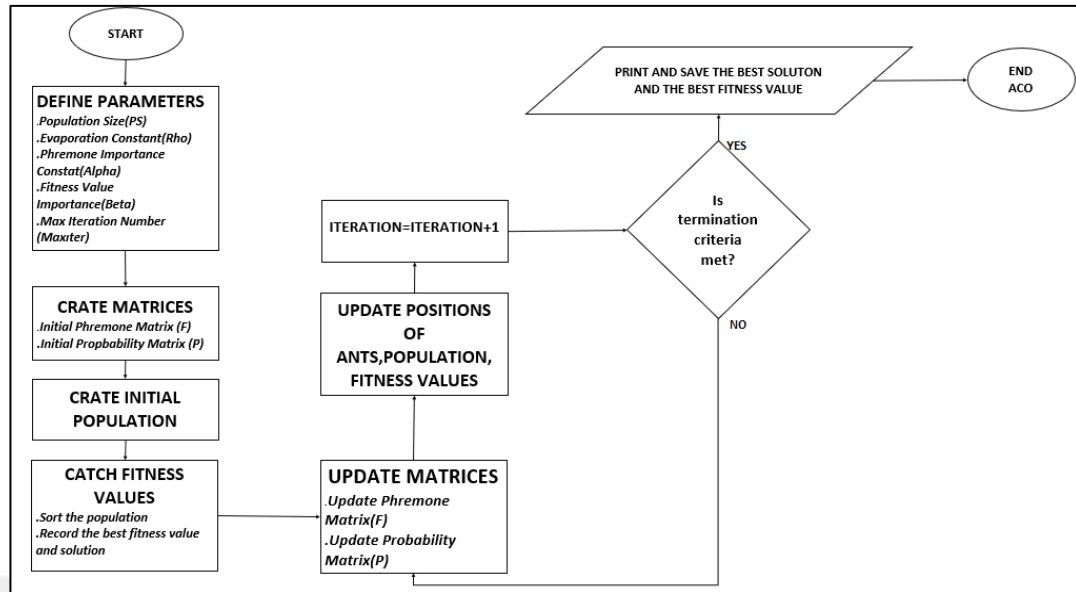


Figure 6.5 Flowchart of ACO

6.2.3 Procedures of Ant Colony Optimization for a Real-Life Bus Scheduling Problem

Each step in the ACO has its own procedure. The ACO algorithm procedures and pseudo codes used in solving the problem that is the subject of this thesis are explained in detail below in table 6.5.

Table 6.5 Pseudo codes of ACO algorithm

```

BEGIN ANT COLONY OPTIMIZATION
SET PARAMETERS PS,RHO,ALPHA,BETA,MaxIter
CREATE MATRICES
  For Var=0 To ScheduleNumber-1
    For Var2=0 To ScheduleNumber-1
      F(Var,Var2)=1
      P(Var,Var2)=1/ScheduleNumber
    Next Var2
  Next Var
CREATE INITIAL POPULATION RANDOMLY
ARRANGE SIMULATION MODEL
CATCH FITNESS VALUES
EVALUATE FITNESS VALUES
  Rank the solutions from the best towards the worst
  For IterationCounter=0 To MaxIter-1
    UPDATE MATRICES
    For Var4=0 To ScheduleNumber-1
       $F(Var3,Var4) = \Delta F(Var3,Var4) + (1 - \text{Rho}) * F(Var3,Var4)$ 
       $P(Var3,Var4) = ((F(Var3,Var4)^{\text{Alpha}}) * P(Var3,Var4)^{\text{Beta}}) / \text{SumRow}(0,Var3)$ 
    Next Var4
    UPDATE POSITIONS
    RE-ARRANGE SIMULATION MODEL
    CATCH FITNESS VALUES
    EVALUATE FITNESS VALUES
    UPDATE THE ITERATION COUNTER
    IterationCounter=IterationCounter+1
    CHECK TERMINATION COUNTER
  Next IterationCounter
REPORT THE RESULT AND END ANT COLONY OPTIMIZATION

```

Codes are written in ARENA simulation software's VBA environment. Thus, the integrated run of the optimization codes and the simulation model is ensured. Written optimization codes are integrated with both the weekday model and the weekend model. Procedures of ACO are explained below.

- **Parameter definition procedure**

PS, *Rho*, *Alpha* and *Beta* parameters are very important parameters for the ACO algorithm. *PS* parameter defines the number of ants in the algorithm. *Rho* parameter is about pheromone evaporation. While *Alpha* constant defines the importance of

pheromone amount in ants' path selection, *Beta* constant defines the importance of cumulative probabilities. There is no specific combination in the literature for these parameters that directly affect the success of the algorithm. The value to be given to any parameter varies according to the dynamics of the problem. Therefore, 27 different combinations are used in the optimization process with the GA. The used combinations are given in table 6.6 below.

Table 6.6 Parameter combinations for ant colony algorithm

	PS	Alpha	Beta
<i>Combination-1</i>	20	0.5	0.5
<i>Combination-2</i>	20	1	0.5
<i>Combination-3</i>	20	1.5	0.5
<i>Combination-4</i>	20	0.5	1
<i>Combination-5</i>	20	1	1
<i>Combination-6</i>	20	1.5	1
<i>Combination-7</i>	20	0.5	1.5
<i>Combination-8</i>	20	1	1.5
<i>Combination-9</i>	20	1.5	1.5
<i>Combination-10</i>	40	0.5	0.5
<i>Combination-11</i>	40	1	0.5
<i>Combination-12</i>	40	1.5	0.5
<i>Combination-13</i>	40	0.5	1
<i>Combination-14</i>	40	1	1
<i>Combination-15</i>	40	1.5	1
<i>Combination-16</i>	40	0.5	1.5
<i>Combination-17</i>	40	1	1.5
<i>Combination-18</i>	40	1.5	1.5
<i>Combination-19</i>	100	0.5	0.5
<i>Combination-20</i>	100	1	0.5
<i>Combination-21</i>	100	1.5	0.5
<i>Combination-22</i>	100	0.5	1
<i>Combination-23</i>	100	1	1
<i>Combination-24</i>	100	1.5	1
<i>Combination-25</i>	100	0.5	1.5
<i>Combination-26</i>	100	1	1.5
<i>Combination-27</i>	100	1.5	1.5

- ***Initial pheromone and initial probability matrices creation procedure***

One of the initial steps of the ACO algorithm is to generate initial pheromone and initial probability matrices. The pheromone matrix is a matrix that shows the superiority of one schedule over another. This matrix includes the pheromone values of nine different schedules available. The more a schedule is used throughout the algorithm, the greater the amount of pheromones. At the beginning of the algorithm, the pheromone amounts of all the schedules are determined as one, since no schedule is superior to the other. The pheromone matrices are updated in each iteration. The probability matrix shows the probability of a schedule being selected by an ant. Probability values of all schedules are given equally, since one schedule does not have superiority over the other at the initial stage. Initial probability values are determined as $1/9$ for all schedules.

- ***Creating initial population***

The ants in the ACO algorithm come together to form the population. The words population or colony refers to the same situation. At the beginning of the ACO algorithm, ants are produced as much as the value of the PS parameter. These ants are randomly assigned to certain positions. The starting positions are determined completely randomly. Random numbers have upper and lower limit. The lower limit is 1 and the upper limit is 9 for random integer numbers during the initial population creation. After initial population has created, simulation model arranged and run.

- ***Fitness value evaluation procedure***

After the initial population has formed, the simulation model has run. Thus, fitness values are caught. After fitness values are caught, population is ranked by fitness values from the best to the worst. Thus, the best solution is determined in the population. The best fitness value is recorded in the algorithm. To catch fitness values simulation model must be run. In the VBA environment of ARENA simulation software there are some special commands. Once simulation model starts to run, codes arrange simulation model and position the ants. Thus, the algorithm generates the new schedule. After simulation model arrangement, the simulation model runs and generates the fitness values. Fitness value is the average travelling time.

- ***Pheromone and probability matrices updating procedure***

Initial pheromone and initial probability matrices are generated at the beginning of the algorithm. The pheromone matrix and probability matrix are updated in each iteration. The pheromone matrix update process is updated according to the frequency with which buses use a schedule. If a bus uses a schedule frequently and has a better average travel time than the others, the pheromone value of that schedule is increased. To update the pheromone matrix, the pheromone exchange matrix must first be calculated. The pheromone exchange matrix is referred to as the Delta F matrix and is denoted as ΔF . The calculation formula (6.1) for the ΔF matrix is given below.

$$\Delta F(x, y) = \Delta F(x, y) + \left(\frac{1}{BestSolution} \right) \quad (6.1)$$

In this thesis, the elite ant method is used for the pheromone update procedure. With this method, the schedules of the ant with the best solution are tried to be transferred to other ants. After the ΔF matrix is created, the pheromone matrix is updated with the help of the following formula (6.2).

$$F(x, y) = \Delta F(x, y) + (1 - Rho) * F(x, y) \quad (6.2)$$

Rho represents the pheromone evaporation coefficient. In this study, the *Rho* value is accepted as 0.05. The *Rho* value allows some of the pheromone amount belonging to a schedule to evaporate in each iteration. Thus, the chance of the algorithm to catch the local minimum is reduced. After updating the pheromone matrix, the probability matrix must be updated. The probability matrix is denoted by *P*. To update the probability matrix, the temporal probability matrix must be calculated. This matrix is denoted by ΔP . The ΔP matrix is calculated with the help of the following formula (6.3).

$$\Delta P(x, y) = F(x, y)^{Alpha} * P(x, y)^{Beta} \quad (6.3)$$

After calculating the ΔP matrix, the row sums of the matrix are taken. The row totals found are called *SumofRow*. The probability matrix *P* is calculated with the help of the following formula (6.4).

$$P(x, y) = \Delta P(x, y) + SumofRow \quad (6.4)$$

- ***Position updating procedure***

The position update procedure is one of the most important procedures of the ACO algorithm. In this procedure, the algorithm uses the knowledge it has learned from past iterations. In this procedure, the ants in the colony update their positions. The goal is to arrive at a better solution. The position update of the ants is done by looking at the pheromone matrix and the probability matrix. In this thesis, the roulette wheel method is used for the position update procedure. This method compares random numbers with values in the previously calculated probability matrix. First, the cumulative sum of the probability matrix is calculated. The algorithm generates random numbers and compares them with the cumulative probability matrix. If the random number generated in the roulette wheel method is less than the value in the cumulative probability matrix, the schedule of the relevant bus is changed. This process is repeated for all schedule values of all ants in the population. Thus, the positions are updated, and the algorithm discovers new solutions.

6.2.4 Results of Ant Colony Optimization

The ACO is run for 27 different parameter combinations. Each ACO iteration includes 20 simulation replications. The results obtained by running the ACO are summarized in the table 6.7 and 6.8 below.

Table 6.8 Results of ACO for the weekend model

[illegible]

ACO run for 27 different parameter combination and provided 12.12% improvement for the weekday model and 11.92% for the weekend model. On contrary to predicted higher population size has not provided better solutions than the others. When alpha parameter equal to one, looks like solutions are better. The best parameter combination for weekend model is 20/1/0.5 while the best parameter combination for weekday model is 100/1/1.5.

6.3 Particle Swarm Optimization

PSO is an algorithm inspired by the movement of bird and fish swarm in the nature. This algorithm aims to achieve optimal or near optimal by mimicking the behavior of swarm of birds and flocks of fish (Kennedy & Eberhart, 1995)

6.3.1 Encoding Process for Particle Swarm Optimization

The encoding process is the process of expressing the real-life problem with the language of the algorithm. The group formed by the birds or fish that exist in the nature is called a swarm. In the PSO algorithm, this community is called swarm. Individuals in the swarm are called particles. Particles come together to form a swarm. The PSO algorithm tracks the movements of particles. In the PSO algorithm, the particles leave their location and discover different areas. Each location that particles discover represents new solutions. The position of a particle has 20 dimensions. These 20 dimensions represent the schedules in the studied problem. Each position represents a solution. There are 20 different values in each position, that is, in each solution. These values refer to the schedules used by the buses. The logic of the PSO algorithm is given in figure 6.6 below.

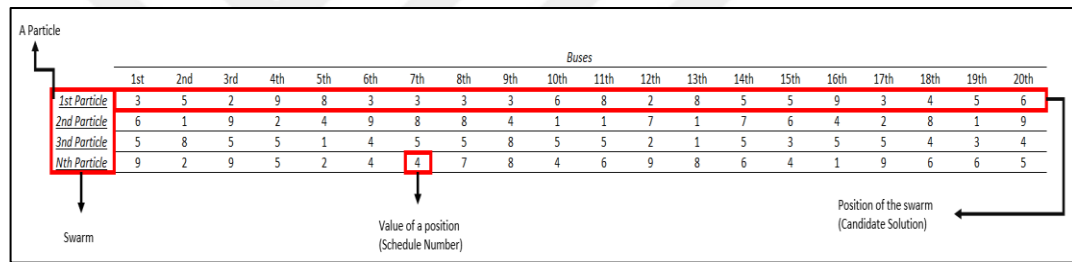


Figure 6.6 Representation of problem with PSO view

Each particle in the PSO algorithm has its own velocity. This velocity is a vectorial velocity. The velocity vector of the particle is recomputed in each iteration. Thanks to velocity vector, the particle leaves its position and goes to new positions. The PSO algorithm is often used for continuous optimization problems. Although the problem discussed in this thesis is a discrete optimization problem, the PSO algorithm has been successfully adapted to this problem.

6.3.2 Flowchart of Particle Swarm Optimization for a Real-Life Bus Scheduling Problem

The flowchart of the algorithm used is given in figure 6.7 has many steps. The PSO algorithm starts with parameter definition. The first swarm is created randomly. After swarm creation, fitness values are calculated. After the velocity vector of the

particles in the swarm is updated, the particles change position. These steps continue until the specified number of iterations is reached.

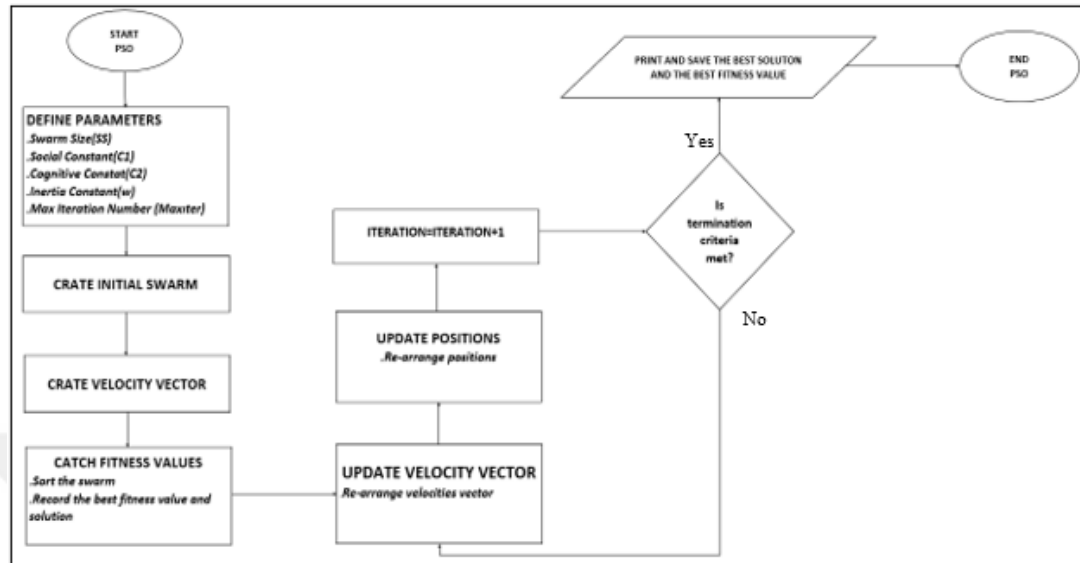


Figure 6.7 Flowchart of PSO

6.3.3 Procedures of Particle Swarm Optimization for a Real-Life Bus Scheduling Problem

Each step in the PSO has its own procedure. The PSO algorithm procedures and pseudo codes used are explained in detail below in table 6.9.

Table 6.9 Pseudo codes of PSO algorithm

BEGIN PARTICLE SWARM OPTIMIZATION ALGORITHM SET PARAMETERS SS,C1,C2,W,MaxIter SET VELOCITY VECTOR GENERATE INITIAL SWARM <i>For IterationCounter=0 To MaxIter</i> ARRANGE SIMULATION MODEL RUN ARRANGED SIMULATION MODEL CATCH FITNESS VALUES EVALUATE FITNESS VALUES <i>Rank the swatm from best towards the worst</i> <i>Save the best</i> UPDATE VELOCITY VECTOR <i>Re-Arrange velocities</i> UPDATE POSITIONS <i>Re-Arrange positions</i> UPDATE THE ITERATION COUNTER <i>IterationCounter=IterationCounter+1</i> CHECK TERMINATION COUNTER <i>Next IterationCounter</i> REPORT THE RESULT AND END PARTICLE SWARM OPTIMIZATION
--

- ***Parameter definition procedure***

SS parameter defines the swarm size. Swarm size has significant effect on success of the algorithm. In addition, the c1, c2 and w parameters are also very important. The C1 coefficient can be expressed as the social coefficient. This coefficient defines the desire of the particles in the algorithm to move towards the particle in the best position in the swarm. The C2 coefficient is the cognitive coefficient. Cognitive coefficient is the desire of a particle in the swarm to go back to its best position in the past. W parameter can be defined as the inertia coefficient. The inertia coefficient expresses the desire of the particle to maintain its position. The values of the parameters in the PSO algorithm change according to the dynamics of the problem. There is no specific combination for these parameters in the literature. For this reason, 27 different combinations are tried. The 27 different combinations tried in the PSO algorithm are given in the table 6.10 below.

Table 6.10 Parameter combinations for PSO

	SS	C1	C2
<i>Combination-1</i>	20	0.5	0.5
<i>Combination-2</i>	20	0.5	1.5
<i>Combination-3</i>	20	0.5	2
<i>Combination-4</i>	20	1.5	0.5
<i>Combination-5</i>	20	1.5	1.5
<i>Combination-6</i>	20	1.5	2
<i>Combination-7</i>	20	2	0.5
<i>Combination-8</i>	20	2	1.5
<i>Combination-9</i>	20	2	2
<i>Combination-10</i>	40	0.5	0.5
<i>Combination-11</i>	40	0.5	1.5
<i>Combination-12</i>	40	0.5	2
<i>Combination-13</i>	40	1.5	0.5
<i>Combination-14</i>	40	1.5	1.5
<i>Combination-15</i>	40	1.5	2
<i>Combination-16</i>	40	2	0.5
<i>Combination-17</i>	40	2	1.5
<i>Combination-18</i>	40	2	2
<i>Combination-19</i>	100	0.5	0.5
<i>Combination-20</i>	100	0.5	1.5
<i>Combination-21</i>	100	0.5	2
<i>Combination-22</i>	100	1.5	0.5
<i>Combination-23</i>	100	1.5	1.5
<i>Combination-24</i>	100	1.5	2
<i>Combination-25</i>	100	2	0.5
<i>Combination-26</i>	100	2	1.5
<i>Combination-27</i>	100	2	2

- ***Velocity vector procedure***

Each particle in the PSO algorithm has a velocity. This velocity is a vectorial velocity. The particles leave the position they are in and go to different positions by using their velocity. The velocity vector defined at the beginning of the algorithm is updated in each iteration. In the initial velocity vector, the velocity of each particle is set to 0.

- ***Creating initial swarm procedure***

One of the steps at the beginning of the PSO algorithm is to define the starting swarm. The number of particles in the initial swarm is equal to the size of the previously determined SS parameter. Particles come together to form a swarm. Each particle consists of 20 different values. These 20 different values represent 20 different buses in a real-life problem. Each value of the particles can take a minimum of 1 and a maximum of 9 integer value. These values represent the schedule numbers used by the buses.

- ***Fitness value evaluation procedure***

After the initial swarm has formed, the simulation model has run. Thus, fitness values are caught. After fitness values are caught, swarm is ranked by fitness values from the best to the worst. Thus, the best solution is determined in the population. The best fitness value is recorded in the algorithm. To catch fitness values simulation model must be run. In the VBA environment of ARENA simulation software there are some special commands. Once simulation model starts to run, codes arrange simulation model and change position of particles. Thus, the algorithm generates the new schedule. After simulation model arrangement, simulation model runs and generates the fitness values. Fitness value is the average travelling time.

- ***Velocity vector updating procedure***

Each of the particles in the swarm has a velocity vector. These velocity vectors are updated every iteration. While updating the velocities, the $c1$, $c2$ and w parameters defined at the beginning of the algorithm are used. In addition to these parameters, the position and current velocity of the particle are also considered. The particle is denoted by P . The parameter $c1$ represents the social coefficient, the parameter $c2$ the cognitive coefficient, and the parameter w represents the inertia coefficient. The velocity vector is denoted as V . The speed update is done with the help of the formula 6.5 below.

$$V(x, y) = (w * V(x, y)) + (c1 * Rnd * (PB(x, y) - P(x, y))) + (c2 * Rnd * (SB(x, y) - P(x, y))) \quad (6.5)$$

Rnd represents a random number in the range 0-1. PB represents the best position of the particle among all past iterations, and SB represents the best position of the swarm among all past iterations. After the velocity vector is updated, it should be arranged. As mentioned before, the values that a particle's position can take are integers between 1 and 9. The maximum speed has been determined so that speeds do not exceed these limits. The maximum speed is calculated with the help of the following formula 6.6.

$$V_{Max}(x, y) = (UpperBound - LowerBound)/2 \quad (6.6)$$

$Upperbound$ is determined as 1 and $Lowerbound$ is determined as 9. The reason for this is that the number of the schedules used by the buses is minimum 1 and maximum 9. If the velocity of a particle exceeds its maximum velocity in the negative direction, the velocity of the particle becomes equal to $-V_{Max}$. However, if the velocity of the particle exceeds the maximum velocity value in the positive direction, the value of the particle becomes equal to the value of V_{Max} . Thus, the updated velocity vector is arranged.

- **Position updating procedure**

After updating the velocity vector, the next step is to update the positions of the particles. Positions are updated according to the particles' velocities and positions. In this step, the algorithm tries to find a better result by using the information learned from the past iterations. The positions of the particles are updated with the help of the formula 6.7 below.

$$P(x, y) = P(x, y) + V(x, y) \quad (6.7)$$

When the positions of the particles and the velocity vector are added together, the new positions of the particles are calculated. However, at this point, care must be taken not to exceed the upper and lower bounds of the value of a position of the particle. Therefore, updated positions must be arranged. If any value at a particle's position falls below the lower bound, that value is set to the lower bound. On the contrary, if a value in the position of a particle goes above the upper bound, that value

becomes equal to the upper bound. Thus, the new positions of the particles are calculated. Each new position represents a candidate solution in the solution space.

6.3.4 Results of Particle Swarm Optimization

The PSO is run for 27 different parameter combinations. Each PSO iteration includes 20 simulation replications. The results obtained by running the PSO are summarized in the table 6.11 and 6.12 below.

Table 6.11 Results of PSO for the weekdays model

[illegible]

Table 6.12 Results of PSO for the weekend model

[illegible]

PSO run for 27 different parameter combination and provided 14.01% improvement for the weekday model and 15.19% for the weekend model. As it predicted, higher population size has provided better solutions than the others. The best parameter combination for weekend model is 100/2/0.5 while the best parameter combination for weekday model is 100/2/0.5.

6.4 Artificial Bee Colony Algorithm

The ABC algorithm is a very powerful algorithm used in solving complex combinatorial optimization problems. The algorithm is inspired by the foraging process of bees in the real world. The algorithm was first introduced by Karaboğa (Karaboğa & Basturk, 2007).

6.4.1 Encoding Process for Artificial Bee Colony Algorithm

The first step to solve the bus scheduling problem using ABC is to express the real-life problem in the language of the ABC. This process is called the encoding process. The real-life system has 20 different buses and 20 different schedules. In this problem, 20 different schedules that already exist are stretched forward and backward for 20 minutes at 5-minute intervals. The schedules of all buses are changed at 5-minute intervals and the best combination is tried to be found. Each combination represents a candidate solution. The ABC algorithm is inspired by the behavior of bees in the food searching process in nature. The bee community is called a colony. There are three types of bees in the colony. Worker bees are the first type of bees. Worker bees are in charge of going to food sources and collecting food. After the worker bees collect enough food from the food source, they return to the hive and transfer the location and quality of the food source to other bees. While transferring information, bees exhibit various dance figures. The location and quality of the food source is transferred to the second type of bees. The second type of bees is called onlooker bees. Onlooker bees wait in the hive and evaluate the information from the worker bees. According to the information received, they go to food sources. The third type of bee is the scout bee. Scout bees are tasked with searching for previously undiscovered food sources. Each food source represents a candidate solution. Thus, the solution space is searched. The quality of food sources is expressed by the fitness value of schedule combinations. The location of the food source represents the schedules.

6.4.2 Flowchart of Artificial Bee Colony Optimization for a Real-Life Bus Scheduling Problem

The ABC algorithm consists of many steps. Each of these steps plays an important role in the success of the algorithm. The algorithm first starts with the definition of colony size. The colony size represents the number of worker bees in the population. Number of worker bees equal to onlooker and scout bees. After the colony created, each member of worker bees assigned to a random food source. Then neighborhood searching process begins. Worker bees evaluate the fitness value and apply greedy selection. This is the first phase of the algorithm. In the second phase worker bees go

back to the hive and transfer the information location and quality of the food source to the onlooker bees. Onlooker bees assigned to the food sources according to the selection probability of food source. Same neighborhood searching process applied for onlooker bees also. In the third and the last phase of algorithm is assigning scout bees to the food sources. The flowchart of the algorithm used is given below in figure 6.8.

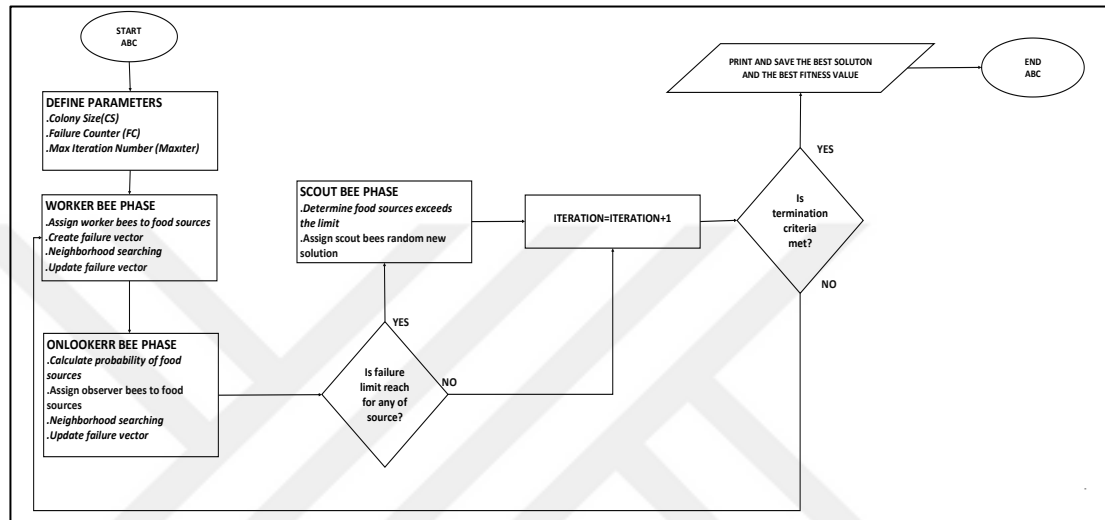


Figure 6.8 Flowchart of ABC algorithm

6.4.3 Procedures of Artificial Bee Colony Algorithm for a Real-Life Bus Scheduling Problem

Each step in the ABC has its own procedure. The ABC algorithm procedures and pseudo codes used in solving the problem are explained in detail below in table 6.13.

Table 6.13 Pseudo codes of ABC algorithm

BEGIN ARTIFICIAL BEE COLONY ALGORITHM	
SET PARAMETERS CS,FailureLimit,MaxIter	
GENERATE INITIAL COLONY	
	<i>For CS=0 to CS-1</i>
	<i>Assign bees to the food sources randomly</i>
	<i>Crate initial failure vector</i>
	<i>Next CS</i>
ARRANGE SIMULATION MODEL	
RUN ARRANGED SIMULATION MODEL	
CATCH FITNESS VALUES	
<i>For IterationCounter=0 to MaxIter</i>	
WORKER BEE PHASE	
	<i>For CS=0 to CS-1</i>
	Determine index for neighborhood search
	Calculate new solutions
	Re-Arrange new solutions
	Re-arrange simulation model
	Catch fitness values for new solution
	Apply greedy selection
	Update failure vector
	<i>Next CS</i>
ONLOOKER BEE PHASE	
	<i>For CS=0 to CS-1</i>
	Calculate selection probabilities for food sources
	Assign onlooker bees to the food sources
	Determine index for neighborhood search
	Calculate new solutions
	Re-Arrange new solutions
	Re-arrange simulation model
	Catch fitness values for new solution
	Apply greedy selection
	Update failure vector
	<i>Next CS</i>
SCOUT BEE PHASE	
	<i>For CS=0 to CS-1</i>
	Determine food sources that exceed failure limit
	Crate new solution
	Replace new solution with the solutions that exceed failure limit
	Re-arrange simulation model
	Catch fitness values for new solution
	<i>Next CS</i>
UPDATE THE ITERATION COUNTER	
	IterationCounter=IterationCounter+1
CHECK TERMINATION COUNTER	
	<i>Next</i>
	<i>MaxIter</i>
REPORT THE RESULT AND END ARTIFICIAL BEE COLONY ALGORITHM	

- ***Parameter definition procedure***

In this procedure, colony size is defined. Colony size represents the number of worker bees. The number of worker bees is equal to the number of onlooker bees and the number of scout bees. Number of worker bees is an important parameter for ABC algorithm. Therefore, three different worker bee numbers are considered. Details are given in table 6.14.

Table 6.14 Parameter combinations for ABC algorithm

	CS
<i>Combination-1</i>	20
<i>Combination-2</i>	40
<i>Combination-3</i>	100

- ***Worker bee phase procedure***

The first step of this procedure is to randomly assign the initially created worker bees to food sources. Each food source represents a candidate solution. After the worker bees are randomly generated, the failure vector is created. The failure vector indicates whether the solution could be improved on each attempt. In this study, the failure limit is taken as 3. Initially, the failure value of each food source is defined as 0. The next step is to assess the quality of food sources. The simulation model is run and the fitness values are obtained. The obtained compliance values are recorded. Then the neighborhood search process begins. Each worker bee searches for a food source close to its location. The fitness values of neighboring solutions are calculated. If the neighboring solution has a better fitness value, the worker bee changes its position. Greedy selection method is used in the solution selection. Thanks to the neighborhood search process, the algorithm performs a local search. If at the end of the neighborhood search process, the fitness value of the neighboring solution is worse than the current solution, the failure value of the relevant solution is increased by one. If the neighboring solution has a better value, the failure value of the relevant solution is reset. This process is repeated for all worker bees. In the neighborhood search process, each worker bee randomly chooses one of the locations where the other worker bees are located. The location of any food source consists of 20 different

values. These 20 different values represent the numbers of the schedules used by the buses. After determining the neighbor to be sought for the worker bees, the value of the neighbor to be sought must be determined. The value to be searched is made randomly, just like the neighbor selection to be searched. During the neighborhood search process, which worker bee will search for which neighbor is obtained with the help of the formula 6.8 below.

$$X_{New}(x, y) = X(x, y) + \emptyset * (X(x, y) - X_p(x, y))$$

$$\emptyset \in [1, -1] \quad (6.8)$$

X_{New} represents the new food source. X represents the location of the current food source, while X_p represents the location of the neighbor to be searched. $\emptyset$ represents a random number in the range +1 -1. In the neighborhood search process, there is a possibility that the food sources will go out of the solution space due to the use of random numbers while creating the newly produced food sources. The newly produced solution must be within the lower and upper limits of the problem. For this reason, newly created resources are checked. If a value of the newly created resource is below the lower limit, that value is set to the lower limit. Conversely, if a value of the newly created resource is above the upper limit, that value is set to the upper limit. In this study, since the values of the sources express the schedules used by the buses, the integer lower limit is 1 and the upper limit is 9.

- ***Onlooker bee phase procedure***

In this phase, the worker bees return to the hive and convey information to the onlooker bees about the location and quality of the food. The step in the algorithm where information flow and learning are provided is the onlooker bee phase. The first step of this phase is to calculate the probability of selection of food sources. Probability values of food sources are calculated with the following formula 6.9.

$$P_i = \frac{fit_i}{\sum fit_i} \quad (6.9)$$

P_i represents the probability value of a food source. The better the fitness value of a food source, the higher the selection probability. fit_i indicates the fitness value of

the i th source. After probability calculation, calculated probability compared with a random number in one and minus one range. If the probability greater than the random number onlooker bees chooses the food source. Then, neighborhood searching process begins. Neighborhood searching process is the same with at worker bee phase. After neighborhood searching process, failure vector updating process applied as well.

- ***Scout bee phase procedure***

In this phase, solutions that cannot improve themselves as a result of neighborhood search are eliminated and other new solutions are produced instead. In this study, the maximum failure limit was determined as 3. In the first step, solutions that exceed the failure limit are determined. In the second stage, new solutions are produced. New solutions are produced with the help of the formula (6.10) below.

$$X_{New}(x, y) = X(x, y) + \emptyset * (X(x, y) - X_p(x, y)) \quad (6.10)$$

After the new solution is produced, onlooker bees turn into worker bees. The fitness values of new solutions are accepted without applying greedy selection.

6.4.4 Results of Artificial Bee Colony Algorithm

The ABC is run for 3 different parameter combinations. Each ABC iteration includes 20 simulation replications. The results obtained by running the ABC are summarized in the table below in table 6.15 and 6.16.

CHAPTER SEVEN

CONCLUSIONS

Buses are frequently used vehicles in public transport. Bus schedules directly affect passenger satisfaction. Non-optimal schedules increase passenger waiting times. Therefore, it directly affects the performance of the system. In addition, bus schedules have a significant effect on the daily gain of the system and the daily cost of the system. Considering that the age we live in is the data era, a better system design by interpreting the collected data has become much more important. For these reasons, bus schedules are extremely important in the transportation industry.

The bus scheduling problem is a problem that seeks answers to multiple questions. Minimizing the travel time of the passengers, reducing the operating costs and increasing the operating income are the main objectives of the bus scheduling problem. The main purpose of the bus scheduling problem discussed in this thesis is to minimize the average travel time of the passengers.

Simulation optimization approach is used in this thesis, which aims to minimize the average travel time of passengers in a real system. First, the simulation of the real system is built. Then, the simulated system is optimized with metaheuristics methods. In the simulation phase, primarily the behavior of the real system is observed. The observed real system behavior is expressed with a process flowchart and the system is modeled in ARENA simulation software according to the process flowchart. Verification and validation processes are carried out to confirm that the modeled system works as desired and reflects the real life system. In the optimization phase, four different algorithms are used. These algorithms are GA, ACO, PSO and ABC algorithms. After the encoding process of the algorithms, the codes are written in VBA environment and integrated into ARENA simulation software. Thus, it is ensured that the optimization algorithm and the simulation model work simultaneously. The performances of the four algorithms used in the optimization phase are compared with each other.

GA run for 27 different parameter combination and provided 12.92% improvement for the weekday model and 14.84% for the weekend model. As it

predicted, higher population size and crossover rate has provided better solutions than the others. The best parameter combination for weekend model is 100/0.8/0.2 while the best parameter combination for weekday model is 100/0.8/0.4.

ACO run for 27 different parameter combination and provided 12.12% improvement for the weekday model and 11.92% for the weekend model. On contrary to predicted higher population size has not provided better solutions than the others. When alpha parameter equal to one, looks like solutions are better. The best parameter combination for weekend model is 20/1/0.5 while the best parameter combination for weekday model is 100/1/1.5.

PSO run for 27 different parameter combination and provided 14.01% improvement for the weekday model and 15.19% for the weekend model. As it predicted, higher population size has provided better solutions than the others. The best parameter combination for weekend model is 100/2/0.5 while the best parameter combination for weekday model is 100/2/0.5.

ABC run for 3 different population size and provided 12.67% improvement for the weekday model and 15.97% for the weekend model. As it predicted, higher population size has provided better solutions than the others. The best population size for weekend model is 100 while the best parameter combination for weekday model is 100.

It has been observed that the results obtained at the end of the optimization phase are better than the current system results.

The calculated average travel time of the passengers in the current system and the best results obtained as a result of optimization methods are given in table 7.1 and table 7.2.

REFERENCES

- Baita, F., Pesenti, R., Ukovich, W., & Favaretto, D. (2000). A comparison of different solution approaches to the vehicle scheduling problem in a practical case. *Computers & Operations Research*, 27(13), 1249-1269. doi: 10.1016/s0305-0548(99)00073-8
- Castelli, L., Pesenti, R., & Ukovich, W. (2004). Scheduling multimodal transportation systems. *European Journal Of Operational Research*, 155(3), 603-615. doi: 10.1016/j.ejor.2003.02.002
- Daduna, J., & Voß, S. (1995). Practical experiences in schedule synchronization. *Lecture Notes In Economics And Mathematical Systems*, 39-55. doi: 10.1007/978-3-642-57762-8_4
- de Palma, A., & Lindsey, R. (2001). Transportation: supply and congestion. *International Encyclopedia Of The Social & Behavioral Sciences*, 15882-15888. doi: 10.1016/b0-08-043076-7/02318-4
- Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant colony optimization. *IEEE Computational Intelligence Magazine*, 1(4), 28-39. doi: 10.1109/ci-m.2006.248054
- Gendreau, M., & Potvin, J. (2019). Handbook of metaheuristics (4th ed.). New York: Springer.
- Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal Of Global Optimization*, 39(3), 459-471. doi: 10.1007/s10898-007-9149-x
- Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings Of ICNN'95 - International Conference On Neural Networks*, 1942-12948. doi: 10.1109/icnn.1995.488968

- Hassold, S., & Ceder, A. (2013). Public transport time-tabling based on maximum-load points using multisize vehicles. *Transportation Research Record: Journal Of The Transportation Research Board*, 2352(1), 104-113. doi: 10.3141/2352-12
- Gkiotsalitis, K. (2020). Bus scheduling considering trip-varying travel times, vehicle availability and capacity. *IET Intelligent Transport Systems*, 14(12), 1594-1605. doi: 10.1049/iet-its.2019.0725
- Kim, B., Kim, S., & Park, J. (2012). A school bus scheduling problem. *European Journal Of Operational Research*, 218(2), 577-585. doi: 10.1016/j.ejor.2011.11.035
- Klemmt, W., & Stemme, W. (1988). Schedule synchronization for public transit networks. *Computer-Aided Transit Scheduling*, 327-335. doi: 10.1007/978-3-642-85966-3_28
- Li, Z., Lam, W., Wong, S., & Sumalee, A. (2010). An activity-based approach for scheduling multimodal transit services. *Transportation*, 37(5), 751-774. doi: 10.1007/s11116-010-9291-z
- Liebchen, C., & Stiller, S. (2008). Delay resistant timetabling. *Public Transport*, 1(1), 55-72. doi: 10.1007/s12469-008-0004-3
- Öner, N., Gultekin, H., & Koç, Ç. (2020). The airport shuttle bus scheduling problem. *International Journal Of Production Research*, 1-23. doi: 10.1080/00207543.2020.1841317
- Pekel, E., & Kara, S. (2015). A Simulation based scheduling in bus rapid transit system. *Sigma Journal Of Engineering And Natural Sciences*, (33), 66-76.
- Pegden, C., Sadowski, R., & Shannon, R. (1995). Introduction to simulation using Siman (5th ed.). New York: McGraw-Hill.
- Rardin, R. (2000). *Optimization in operations research*. Upper Saddle River (New Jersey): Prentice Hall.

- Saha, J. (1970). An Algorithm for Bus Scheduling Problems. *Journal Of The Operational Research Society*, 21(4), 463-474. doi: 10.1057/jors.1970.95
- Shariat, A., & Amiripour, S. (2019). Creating bus timetables under sthochastic demand. *International Journal of Industrial Engineering And Production Research (IJIE) (English)*, 20(3), 83-91.
- Wang, H., & Shen, J. (2007). Heuristic approaches for solving transit vehicle scheduling problem with route and fueling time constraints. *Applied Mathematics And Computation*, 190(2), 1237-1249. doi: 10.1016/j.amc.2007.02.141
- Wei, M., Chen, X., Sun, B., & Zhu, Y. (2015). Model and algorithm for resolving regional bus scheduling problems with fuzzy travel times. *Journal Of Intelligent & Fuzzy Systems*, 29(6), 2689-2696. doi: 10.3233/ifs-151972
- Wei, M., Sun, B., & Jin, W. (2012). Model and algorithm of regional bus scheduling with grey travel time. *Journal Of Transportation Systems Engineering And Information Technology*, 12(6), 106-112. doi: 10.1016/s1570-6672(11)60237-7
- Wihartiko, F., Buono, A., & Silalahi, B. (2017). Integer programming model for optimizing bus timetable using genetic algorithm. *IOP Conference Series: Materials Science And Engineering*, 166, 012016. doi: 10.1088/1757-899x/166/1/012016
- Yalçınkaya, Ö., & Mirac Bayhan, G. (2012). A feasible timetable generator simulation modelling framework for train scheduling problem. *Simulation Modelling Practice And Theory*, 20(1), 124-141. doi: 10.1016/j.simpat.2011.09.005
- Zhoucong, X., Peijia, H., Jing, T., & Liping, L. (2013). Transit vehicles intelligent scheduling optimization based on the division of characteristic periods. *Procedia - Social And Behavioral Sciences*, 96, 1502-1512. doi: 10.1016/j.sbspro.2013.08.171

APPENDICES

APPENDIX - 1 : Current schedule for route 1

Route Number - 1	
From First Bus Stop	From the Mall
11:00	11:30
12:00	12:30
13:10	13:50
14:40	15:10
15:50	16:30
17:10	17:50
18:30	19:10
19:50	20:30
21:00	21:30

APPENDIX - 2 : Current schedule for route 2

Route Number - 2	
From First Bus Stop	From the Mall
10:30	11:00
11:30	12:00
11:50	12:30
12:20	13:00
13:00	13:30
13:30	14:00
14:00	14:30
14:30	15:00
15:00	15:30
15:30	16:00
16:00	16:30
16:30	17:00
17:00	17:30
17:30	18:00
18:00	18:30
18:30	19:00
19:00	19:30
19:30	20:00
20:00	21:00
20:45	21:30

APPENDIX - 3: Current schedule for route 3

Route Number - 3	
From First Bus Stop	From the Mall
10:30	11:00
11:00	11:30
11:30	12:00
11:50	12:30
12:20	13:00
13:00	13:30
13:30	14:00
14:00	14:30
14:30	15:00
15:00	15:30
15:30	16:00
16:00	16:30
16:30	17:00
17:00	17:30
17:30	18:00
18:00	18:30
18:30	19:00
19:00	19:30
19:30	20:00
20:30	21:00

APPENDIX - 4: Current schedule for route 4

Route Number - 4	
From First Bus Stop	From the Mall
10:00	10:30
10:30	11:00
10:45	11:15
11:00	11:30
11:15	11:45
11:30	12:00
11:45	12:30
12:00	13:00
12:15	13:15
12:30	13:30
13:00	14:00
13:30	14:15
13:45	14:30
14:00	14:45
14:30	15:00
14:45	15:30
15:00	15:45
15:15	16:00
15:30	16:15
16:00	16:30
16:15	16:45
16:30	17:00
16:45	17:15
17:00	17:30
17:15	18:00
17:30	18:15
17:45	18:30
18:00	18:45
18:30	19:00
18:45	19:15
19:00	19:30
19:15	19:45
19:30	20:00
19:45	20:15
20:00	20:30
20:15	20:45
20:30	21:00
20:45	21:15
21:00	21:30
21:15	21:45

APPENDIX - 5: Current schedule for route 5

Route Number - 5	
From First Bus Stop	From the Mall
11:00	11:30
12:00	12:30
12:50	13:30
14:00	14:30
15:00	15:30
16:00	16:30
17:00	17:30
18:30	19:00
19:30	20:00

APPENDIX - 6: Current schedule for route 6

Route Number - 6	
From First Bus Stop	From the Mall
11:00	11:30
11:50	12:30
13:00	13:30
14:00	14:30
15:00	15:30
16:00	16:30
17:00	17:30
18:00	18:30
19:00	19:30
20:00	20:30

APPENDIX - 7: Current schedule for route 7

Route Number - 7	
From First Bus Stop	From the Mall
11:00	11:30
11:30	11:50
11:50	12:30
12:20	13:00
13:00	13:30
13:30	14:00
14:00	14:30
14:30	15:00
15:00	15:30
15:30	16:00
16:00	16:30
16:30	17:00
17:00	17:30
17:30	18:00
18:00	18:30
18:30	19:00
19:00	19:30
19:30	20:00
20:00	20:30
20:30	21:00

APPENDIX - 8: Current schedule for route 8

Route Number - 8	
From First Bus Stop	From the Mall
10:00	10:40
11:20	12:00
12:30	13:10
13:50	14:30
15:00	15:30
16:00	16:50
17:30	18:10
18:50	19:30
20:00	20:30
21:00	21:30

APPENDIX - 9: Current schedule for route 9

Route Number - 9	
From First Bus Stop	From the Mall
12:00	12:30
12:50	13:30
14:00	14:30
15:00	15:30
16:00	16:30
16:50	17:05

APPENDIX - 10: Current schedule for route 10

Route Number - 10	
From First Bus Stop	From the Mall
10:30	11:10
10:50	11:30
11:10	11:50
11:50	12:30
12:05	12:45
12:30	13:10
13:10	13:50
13:25	14:10
13:50	14:30
14:30	15:10
14:50	15:30
15:10	15:50
15:50	16:30
16:00	16:45
16:30	17:10
17:10	17:50
17:25	18:10
17:50	18:30
18:30	19:10
18:50	19:20
19:10	19:50
19:50	20:20
20:00	20:30
20:30	21:00
20:50	21:15
21:00	21:30
21:30	21:50

APPENDIX - 11: Current schedule for route 11

Route Number - 11	
From First Bus Stop	From the Mall
11:00	11:30
11:30	12:00
11:50	12:30
12:20	13:00
13:00	13:30
13:30	14:00
14:00	14:30
14:30	15:00
15:00	15:30
15:30	16:00
16:00	16:30
16:30	17:00
17:00	17:30
17:30	18:00
18:00	18:30
18:30	19:00
19:00	19:30
19:30	20:00
20:00	20:30
20:30	21:00
21:00	21:30

APPENDIX - 12: Optimal schedule of weekdays model for route 1

Route Number - 1	
From First Bus Stop	From the Mall
11:20	11:50
12:20	12:50
13:30	14:10
15:00	15:30
16:10	16:50
17:30	18:10
18:50	19:30
20:10	20:50
21:20	21:50

APPENDIX - 13: Optimal schedule of weekdays model for route 2

Route Number - 2	
From First Bus Stop	From the Mall
10:30	11:00
11:35	12:05
11:50	12:30
12:25	13:05
13:00	13:30
13:35	14:05
14:00	14:30
14:35	15:05
15:00	15:30
15:35	16:05
16:00	16:30
16:35	17:05
17:00	17:30
17:35	18:05
18:00	18:30
18:35	19:05
19:00	19:30
19:35	20:05
20:00	21:00
20:50	21:35

APPENDIX - 14: Optimal schedule of weekdays model for route 3

Route Number - 3	
From First Bus Stop	From the Mall
10:50	11:20
10:45	11:15
11:50	12:20
11:35	12:15
12:40	13:20
12:45	13:15
13:50	14:20
13:45	14:15
14:50	15:20
14:45	15:15
15:50	16:20
15:45	16:15
16:50	17:20
16:45	17:15
17:50	18:20
17:45	18:15
18:50	19:20
18:45	19:15
19:50	20:20
20:15	20:45

APPENDIX - 15: Optimal schedule of weekdays model for route 4

Route Number - 4	
From First Bus Stop	From the Mall
10:20	10:50
10:35	11:05
11:05	11:35
11:20	11:50
11:35	12:05
11:35	12:05
12:05	12:50
12:20	13:20
12:35	13:35
12:35	13:35
13:20	14:20
13:50	14:35
14:05	14:50
14:05	14:50
14:50	15:20
15:05	15:50
15:20	16:05
15:20	16:05
15:50	16:35
16:20	16:50
16:35	17:05
16:35	17:05
17:05	17:35
17:20	17:50
17:35	18:20
17:35	18:20
18:05	18:50
18:20	19:05
18:50	19:20
18:50	19:20
19:20	19:50
19:35	20:05
19:50	20:20
19:50	20:20
20:20	20:50
20:35	21:05
20:50	21:20
20:50	21:20
21:20	21:50
21:35	22:05

APPENDIX - 16: Optimal schedule of weekdays model for route 5

Route Number - 5	
From First Bus Stop	From the Mall
10:30	11:00
11:30	12:00
12:20	13:00
13:30	14:00
14:30	15:00
15:30	16:00
16:30	17:00
18:00	18:30
19:00	19:30

APPENDIX - 17: Optimal schedule of weekdays model for route 6

Route Number - 6	
From First Bus Stop	From the Mall
10:45	11:15
11:35	12:15
12:45	13:15
13:45	14:15
14:45	15:15
15:45	16:15
16:45	17:15
17:45	18:15
18:45	19:15
19:45	20:15

APPENDIX - 18: Optimal schedule of weekdays model for route 7

Route Number - 7	
From First Bus Stop	From the Mall
10:30	11:00
11:15	11:35
11:20	12:00
12:05	12:45
12:30	13:00
13:15	13:45
13:30	14:00
14:15	14:45
14:30	15:00
15:15	15:45
15:30	16:00
16:15	16:45
16:30	17:00
17:15	17:45
17:30	18:00
18:15	18:45
18:30	19:00
19:15	19:45
19:30	20:00
20:15	20:45

APPENDIX - 19: Optimal schedule of weekdays model for route 8

Route Number - 8	
From First Bus Stop	From the Mall
09:50	10:30
11:10	11:50
12:20	13:00
13:40	14:20
14:50	15:20
15:00	16:40
17:20	18:00
18:40	19:20
19:50	20:20
20:50	21:20

APPENDIX - 20: Optimal schedule of weekdays model for route 9

Route Number - 9	
From First Bus Stop	From the Mall
12:15	12:45
13:05	13:45
14:15	14:45
15:15	15:45
16:15	16:45
17:05	17:20

APPENDIX - 21: Optimal schedule of weekdays model for route 10

Route Number - 10	
From First Bus Stop	From the Mall
10:00	10:40
10:20	11:00
11:05	11:45
11:20	12:00
11:35	12:15
12:25	13:05
12:40	13:20
12:55	13:40
13:45	14:25
14:00	14:40
14:20	15:00
15:05	15:45
15:20	16:00
15:30	16:15
16:25	17:05
16:40	17:20
16:55	17:40
17:45	18:25
18:00	18:40
18:20	18:50
19:05	19:45
19:20	19:50
19:30	20:00
20:25	20:55
20:20	20:45
20:30	21:00
21:25	21:45

APPENDIX - 22: Optimal schedule of weekdays model for route 11

Route Number - 11	
From First Bus Stop	From the Mall
11:20	11:50
11:45	12:15
12:10	12:50
12:35	13:15
13:20	13:50
13:45	14:15
14:20	14:50
14:45	15:15
15:20	15:50
15:45	16:15
16:20	16:50
16:45	17:15
17:20	17:50
17:45	18:15
18:20	18:50
18:45	19:15
19:20	19:50
19:45	20:15
20:20	20:50
20:45	21:15
21:20	21:50

APPENDIX - 23: Optimal schedule of weekend model for route 1

Route Number - 1	
From First Bus Stop	From the Mall
10:55	11:25
11:55	12:25
13:05	13:45
14:35	15:05
15:45	16:25
17:05	17:45
18:25	19:05
19:45	20:25
20:55	21:25

APPENDIX - 24: Optimal schedule of weekend model for route 2

Route Number - 2	
From First Bus Stop	From the Mall
10:15	10:45
11:30	12:00
11:35	12:15
12:20	13:00
12:45	13:15
13:30	14:00
13:45	14:15
14:30	15:00
14:45	15:15
15:30	16:00
15:45	16:15
16:30	17:00
16:45	17:15
17:30	18:00
17:45	18:15
18:30	19:00
18:45	19:15
19:30	20:00
19:45	20:45
20:45	21:30

APPENDIX - 25: Optimal schedule of weekend model for route 3

Route Number - 3	
From First Bus Stop	From the Mall
10:20	10:50
10:40	11:10
11:20	11:50
11:30	12:10
12:10	12:50
12:40	13:10
13:20	13:50
13:40	14:10
14:20	14:50
14:40	15:10
15:20	15:50
15:40	16:10
16:20	16:50
16:40	17:10
17:20	17:50
17:40	18:10
18:20	18:50
18:40	19:10
19:20	19:50
20:10	20:40

APPENDIX - 26: Optimal schedule of weekend model for route 4

Route Number - 4	
From First Bus Stop	From the Mall
10:20	10:50
10:15	10:45
10:45	11:15
10:50	11:20
11:35	12:05
11:15	11:45
11:45	12:30
11:50	12:50
12:35	13:35
12:15	13:15
13:00	14:00
13:20	14:05
14:05	14:50
13:45	14:30
14:30	15:00
14:35	15:20
15:20	16:05
15:00	15:45
15:30	16:15
15:50	16:20
16:35	17:05
16:15	16:45
16:45	17:15
16:50	17:20
17:35	18:20
17:15	18:00
17:45	18:30
17:50	18:35
18:50	19:20
18:30	19:00
19:00	19:30
19:05	19:35
19:50	20:20
19:30	20:00
20:00	20:30
20:05	20:35
20:50	21:20
20:30	21:00
21:00	21:30
21:05	21:35

APPENDIX - 27: Optimal schedule of weekend model for route 5

Route Number - 5	
From First Bus Stop	From the Mall
11:15	11:45
12:15	12:45
13:05	13:45
14:15	14:45
15:15	15:45
16:15	16:45
17:15	17:45
18:45	19:15
19:45	20:15

APPENDIX - 28: Optimal schedule of weekend model for route 6

Route Number - 6	
From First Bus Stop	From the Mall
11:20	11:50
12:10	12:50
13:20	13:50
14:20	14:50
15:20	15:50
16:20	16:50
17:20	17:50
18:20	18:50
19:20	19:50
20:20	20:50

APPENDIX - 29: Optimal schedule of weekend model for route 7

Route Number - 7	
From First Bus Stop	From the Mall
11:00	11:30
11:40	12:00
11:50	12:30
12:30	13:10
13:00	13:30
13:40	14:10
14:00	14:30
14:40	15:10
15:00	15:30
15:40	16:10
16:00	16:30
16:40	17:10
17:00	17:30
17:40	18:10
18:00	18:30
18:40	19:10
19:00	19:30
19:40	20:10
20:00	20:30
20:40	21:10

APPENDIX - 30: Optimal schedule of weekend model for route 8

Route Number - 8	
From First Bus Stop	From the Mall
09:50	10:30
11:10	11:50
12:20	13:00
13:40	14:20
14:50	15:20
15:50	16:40
17:20	18:00
18:40	19:20
19:50	20:20
20:50	21:20

APPENDIX - 31: Optimal schedule of weekend model for route 9

Route Number - 9	
From First Bus Stop	From the Mall
11:40	12:10
12:30	13:10
13:40	14:10
14:40	15:10
15:40	16:10
16:30	16:45

APPENDIX - 32: Optimal schedule of weekend model for route 10

Route Number - 10	
From First Bus Stop	From the Mall
10:15	10:55
10:40	11:20
11:15	11:55
11:35	12:15
11:55	12:35
12:35	13:15
12:55	13:35
13:15	14:00
13:55	14:35
14:15	14:55
14:40	15:20
15:15	15:55
15:35	16:15
15:50	16:35
16:35	17:15
16:55	17:35
17:15	18:00
17:55	18:35
18:15	18:55
18:40	19:10
19:15	19:55
19:35	20:05
19:50	20:20
20:35	21:05
20:35	21:00
20:50	21:20
21:35	21:55

APPENDIX - 33: Optimal schedule of weekend model for route 11

Route Number - 11	
From First Bus Stop	From the Mall
11:10	11:40
11:30	12:00
12:00	12:40
12:20	13:00
13:10	13:40
13:30	14:00
14:10	14:40
14:30	15:00
15:10	15:40
15:30	16:00
16:10	16:40
16:30	17:00
17:10	17:40
17:30	18:00
18:10	18:40
18:30	19:00
19:10	19:40
19:30	20:00
20:10	20:40
20:30	21:00
21:10	21:40

APPENDIX - 34: Statistical Distributions for passenger arrivals

Bus Stop #	Bus Stop Name	Route	Weekdays Model Slot1	Weekdays Model Slot2	Weekdays Model Slot3	Weekend Model Slot1	Weekend Model Slot2	Weekend Model Slot3
1	Evka5 Toki	Evka-5	16 + EXPO(14.9)	8 + EXPO(6.15)	9.5 + EXPO(10.6)	5+ EXPO(21)	13+EXPO(24)	6 + EXPO(33.7)
2	Evka5 Toki Yolu Girişi	Evka-5	26 + ERLA(14.8, 1)	NORM(25.8, 8.27)	NORM(28.8,14.27)	9+ EXPO(35)	14+EXPO(25)	30+EXPO(40.5)
3	Köşe-2 Durağı	Evka-5	NORM(35.3, 7.17)	21 + 32 * BETA(0.94, 1.07)	21+32 * BETA(6.94,8.07)	NORM(22.7, 9.19)	21+WEIB(11.5,2.24)	25+EXPO(25)
4	Pazaryeri(cami)	Evka-5	NORM(35.3, 7.17)	NORM(31.4, 6.33)	NORM(35.4,12.33)	NORM(21.6, 9.19)	12+EXPO(24)	23 + WEIB(46.5, 0.511)
5	Tansaş Trafo Nokta	Evka-5	TRIA(30, 39, 60)	20 + EXPO(34.9)	24+ EXPO(39.9)	TRIA(20,29, 34)	NORM(36.1, 9.56)	24+EXPO(41.2)
6	Postahane	Evka-5	TRIA(30, 39, 60)	20 + EXPO(22.6)	23+ EXPO(26.6)	12 + GAMM(57.4, 0.263)	15 + WEIB(24.8, 0.626)	20+EXPO(39.3)
7	Atatürk Mahallesi	Evka-5	TRIA(30, 45, 60)	20 + EXPO(15.8)	22+ EXPO(21.8)	20+ EXPO(19.2)	23+EXPO(30)	23+EXPO(25.1)
8	İlk Evler	Evka-5	30 + WEIB(25.2, 3.46)	20 + WEIB(39.2, 1.55)	34+WEIB(44.2,9.55)	23+ WEIB(23,8)	17 + EXPO(41.4)	19 + WEIB(67.1, 0.665)
9	Park Manav İMKB Lisesi	Evka-5	TRIA(30, 33, 60)	20 + GAMM(15.8, 2.95)	21+GAMM(18.8,5.95)	28+ EXPO(16)	24+EXPO(25)	24+EXPO(19)
10	Otobüs Son Durak	Evka-2	TRIA(30, 39, 60)	38 + EXPO(9.16)	40+ EXPO(13.16)	21+GAMM(57.2,8.78)	15.5 + 85 * BETA(0.4, 0.482)	30+EXPO(20)
11	Manav Durağı	Evka-2	NORM(36.1, 9.56)	20 + EXPO(16)	26+ EXPO(21)	NORM(36.1, 9.56)	12+EXPO(25)	26 + WEIB(48.1, 0.479)
12	Dede durağı	Evka-2	TRIA(30, 45, 60)	20 + GAMM(79.7, 0.583)	28+GAMM(87.7,7.583)	17+WEIB(24.2,12.55)	19+EXPO(22)	NORM(146, 80.1)
13	Eski Minibüs Son durağı	Evka-2	NORM(47.1, 5.25)	20 + EXPO(22.5)	23+ EXPO(26.5)	NORM(47.1, 5.25)	31+WEIB(18.2,4.44)	21 + WEIB(49.4, 0.817)
14	Milli Piyango	Evka-2	30 + WEIB(0.0673, 0.206)	20 + EXPO(17.2)	23+ EXPO(23.2)	29+EXPO(33)	11+EXPO(16)	19 + WEIB(37.9, 0.515)
15	Cami_Evka2	Evka-2	NORM(33.2, 4.99)	20 + EXPO(19.8)	24+ EXPO(25.8)	NORM(33.2, 4.99)	23 + 24 * BETA(0.64, 1.12)	19 + WEIB(37.9, 0.515)
16	Manavlar	Evka-2	30 + 30 * BETA(2.34, 1.41)	18 + LOGN(17.7,7)	21+ WEIB(1.84, 0.21)	23+EXPO(31)	14+EXPO(22)	24+EXPO(20)
17	Telekom Durağı	Evka-2	NORM(38.8, 9.62)	18 + EXPO(26.4)	22+ EXPO(30.4)	NORM(38.8, 9.62)	25+EXPO(27)	25+EXPO(30.4)

APPENDIX - 34: Statistical Distributions for passenger arrivals (cont.)

Bus Stop #	Bus Stop Name	Route	Weekdays Model Slot1	Weekdays Model Slot2	Weekdays Model Slot3	Weekend Model Slot1	Weekend Model Slot2	Weekend Model Slot3
18	Kömürcü	Evka-2	NORM(40.7, 7.62)	TRIA(23, 47.4, 150)	19+ WEIB(1.22, 0.11)	NORM(40.7, 7.62)	16 + ERLA(29.1, 1)	24+EXPO(32)
19	Dostlar	Evka-2	UNIF(30, 60)	22 + EXPO(22)	25+ EXPO(24)	21+LOGN(15.2,16.1)	15 + EXPO(28.6)	31+EXPO(42.2)
20	Cami-2	Evka-2	TRIA(30, 31.5, 45)	22 + EXPO(19.2)	28+ EXPO(22.2)	34+EXPO(30)	13+EXPO(27)	30+EXPO(32)
21	Aydınlık Evler	Evka-2	30 + WEIB(1.32, 0.25)	22 + EXPO(24.1)	27+ EXPO(30.1)	34+EXPO(21)	15+EXPO(22)	32+EXPO(45)
22	Oyak	Evka-2	30 + LOGN(7.21, 2.75)	22 + LOGN(19.6, 10.4)	17 +LOGN(29.6,14.4)	21+WEIB(44.1,11.20)	13+ LOGN(24.9,27.7)	24+EXPO(41)
23	Karpuzcu	Evka-2	30 + WEIB(3.83, 0.31)	22 + LOGN(22.9, 30.7)	17+ LOGN(25.9,37.7)	25+EXPO(25)	15+EXPO(24)	23+EXPO(22)
24	İzkent Tepe Durağı	Egekent	30 + EXPO(11)	22 + EXPO(11.5)	27+ EXPO(17.5)	16+ EXPO(4)	23+EXPO(14)	29+EXPO(34)
25	Meydan	Egekent	30 + LOGN(9.85, 4.71)	22 + EXPO(12.7)	28+ EXPO(18.7)	24+EXPO(23)	17+EXPO(22)	32+EXPO(43.4)
26	Halısaha	Egekent	30 + WEIB(0.0344, 0.225)	22 + EXPO(16.1)	28+ EXPO(22.1)	26+EXPO(18)	12+EXPO(25)	19 + EXPO(61.8)
27	Tansaş	Egekent	NORM(37, 6.16)	22 + EXPO(16.4)	25+ EXPO(22.4)	NORM(37, 6.16)	16+EXPO(14)	23+EXPO(34)
28	Eshotçular	Egekent	NORM(34.7, 6.51)	22 + EXPO(16.8)	30 + WEIB(21.8, 0.429)	NORM(34.7, 6.51)	11+EXPO(31)	29+EXPO(23)
29	Adliyeciler	Egekent	30 + WEIB(0.109, 0.217)	18 + EXPO(22.7)	20+ EXPO(25.7)	26+EXPO(32)	23+EXPO(13)	19 + WEIB(53.7, 0.499)
30	Trafo	Egekent	30 + ERLA(1, 1)	14 + EXPO(17.5)	16+ EXPO(22.5)	16+WEIB(38,9.95)	21+EXPO(25)	25+EXPO(42)
31	Postacılar	Egekent	30 + WEIB(0.179, 0.218)	11.5 + 79 * BETA(0.418, 1.14)	9.79 + 56 * BETA(0.721, 1.58)	22+EXPO(15)	15+EXPO(16)	21+EXPO(45)
32	Köşe	Egekent	30 + EXPO(9.64)	18 + EXPO(7.62)	22+ EXPO(13.62)	18+ EXPO(6.64)	25+EXPO(19)	29+EXPO(20.1)
33	Ormancı Durağı	Egekent	30 + WEIB(12.6, 2.89)	18 + LOGN(18.2, 18.1)	21+LOGN(25.2,26.1)	26+EXPO(17)	19+EXPO(29)	25+EXPO(18.9)
34	Egekent Girişi	Egekent	25 + WEIB(56.5, 0.664)	NORM(27.1, 10.6)	NORM(34.1,21.6)	32+EXPO(13)	20+EXPO(18)	NORM(93.3, 54.2)
35	Çiğli Kahveler	Egekent	25 + WEIB(15.3, 0.324)	18 + EXPO(13)	21+ EXPO(15)	17+LOGN(11,8.2)	21+ LOGN(21.6,12.1)	25+EXPO(30.3)

APPENDIX - 34: Statistical Distributions for passenger arrivals (cont.)

Bus Stop #	Bus Stop Name	Route	Weekdays Model Slot1	Weekdays Model Slot2	Weekdays Model Slot3	Weekend Model Slot1	Weekend Model Slot2	Weekend Model Slot3
36	İtfaiye	Egekent	25 + EXPO(53.5)	18 + 58 * BETA(1.48, 2.94)	58 * BETA(11.48,9.94)	31+EXPO(24)	14+EXPO(20)	23+EXPO(32.4)
37	Tersane	Tersane	2 + EXPO(7.7)	8 + EXPO(4.2)	3+ EXPO(7.2)	10+ EXPO(22.7)	24+EXPO(26)	26+EXPO(32.4)
38	Selçuk Yaşar İÖO	Tersane	26 + EXPO(20.3)	18 + EXPO(10.8)	20+ EXPO(16.8)	17+ EXPO(15.3)	22+EXPO(30)	21+EXPO(18.6)
39	Alaybey Eski Tansaş	Tersane	35 + EXPO(4.86)	NORM(36.3, 13.2)	NORM(39.3,17.2)	27+ EXPO(18)	12+EXPO(16)	26 + WEIB(36.9, 0.396)
40	Yeni GİRNE	Yeni_Girne	26 + EXPO(23.5)	18 + WEIB(21.7, 1.65)	22+WEIB(31.7,10.65)	16+ EXPO(18.5)	21+EXPO(16)	22+EXPO(40)
41	Postacılar Mahallesi	Yeni_Girne	NORM(46.2, 12.9)	18 + EXPO(15)	22+ EXPO(39)	NORM(46.2, 12.9)	NORM(24.4,16.2)	31+EXPO(21.2)
42	Yeni GİRNE Caddesi	Yeni_Girne	30 + WEIB(19.6, 1.52)	18 + EXPO(20.9)	24+ EXPO(31)	16+ EXPO(18.5)	18+EXPO(30)	28+EXPO(30.3)
43	Dedebaşı	Yeni_Girne	NORM(36.8, 6.3)	18 + ERLA(15.6, 1)	26 + WEIB(21.1, 0.377)	NORM(36.8, 6.3)	15 + WEIB(24.8, 0.906)	19 + EXPO(71.3)
44	Karşıyaka Devlet Hastanesi	Yeni_Girne	30 + WEIB(17.3, 1.33)	NORM(27.3, 10.7)	24 + WEIB(12.6, 1.19)	16+ EXPO(18.5)	13+EXPO(13)	21 + WEIB(32.9, 0.519)
45	Serinkuyu	Yeni_Girne	30 + ERLA(15.4, 1)	18 + ERLA(18.7, 1)	26 + WEIB(19.9, 0.372)	16+ EXPO(18.5)	10+EXPO(30)	31+EXPO(24)
46	Çeşme	Yeni_Girne	30 + 60 * BETA(1.55, 2.58)	18 + EXPO(17.2)	24+ EXPO(22.2)	NORM(31.8, 3.3)	15 + EXPO(18.7)	24+EXPO(35)
47	Jandarma	Yeni_Girne	30 + EXPO(12.7)	18 + GAMM(16.6, 0.968)	30+GAMM(27.6,9.968)	16+ EXPO(18.5)	17+EXPO(21)	25+EXPO(33)
48	Benzinlik	Yeni_Girne	30 + WEIB(39.7, 1.56)	18 + LOGN(15.5, 7.97)	25+LOGN(20.5,11.97)	15+LOGN(15.2,8.2)	22+WEIB(18,6.1)	30 + EXPO(58)
49	Ayyıldız	Yeni_Girne	30 + WEIB(0.228, 0.237)	18 + 78 * BETA(2.17, 9.03)	78 * BETA(9.17,14.03)	16+ EXPO(18.5)	24+EXPO(24)	23+EXPO(27.5)

APPENDIX - 34: Statistical Distributions for passenger arrivals (cont.)

Bus Stop #	Bus Stop Name	Route	Weekdays Model Slot1	Weekdays Model Slot2	Weekdays Model Slot3	Weekend Model Slot1	Weekend Model Slot2	Weekend Model Slot3
50	Anadolu Lisesi	Yeni_Girne	50 + WEIB(25.7, 1.61)	18 + 78 * BETA(0.817, 2.25)	78 * BETA(4.81,5.25)	16+ EXPO(18.5)	22+EXPO(29)	21+EXPO(18)
51	Bostanlı İskele	Bostanlı	30 + EXPO(32.1)	TRIA(18, 62.1, 90)	30 + EXPO(43.1)	19+ EXPO(28.1)	20+EXPO(16)	29+EXPO(30)
52	Atatürk Spor Salonu	Bostanlı	26 + ERLA(22.5, 1)	NORM(28.6, 15)	30 + EXPO(30.9)	16+ EXPO(18.5)	14+EXPO(28)	23 + WEIB(56.5, 0.696)
53	EgePark	Bostanlı	TRIA(26, 31.3, 90)	NORM(30.6, 17.8)	NORM(39.6,26.8)	19+WEIB(39,6.95)	11+WEIB(10,8.44)	29+EXPO(24)
54	Süleyman Demirel Lisesi	Bostanlı	26 + LOGN(8.33, 6.1)	NORM(29.6, 13)	26 + EXPO(60.9)	16+ EXPO(18.5)	10+EXPO(28)	20+EXPO(34)
55	Yonca Durağı	Esin_Sitesi	26 + EXPO(14.9)	18 + LOGN(21.9, 10)	26 + EXPO(44.4)	15+ EXPO(11.9)	20+EXPO(22)	32+EXPO(37)
56	Esin Sit. Son Durak	Esin_Sitesi	26 + EXPO(21.5)	18 + WEIB(27.6, 0.976)	29 +WEIB(38.6,3.976)	17+ EXPO(14.5)	10+EXPO(16)	19 + EXPO(22)
57	Körfezköy	Esin_Sitesi	29.5 + 61 * BETA(0.417, 0.627)	TRIA(18, 62.1, 90)	26 + EXPO(58.4)	16+ EXPO(18.5)	13+EXPO(29)	26+EXPO(25)
58	Körfez Evleri	Esin_Sitesi	25 + WEIB(42.9, 0.586)	18 + WEIB(23.2, 1.36)	28+ WEIB(31.2,9.36)	34+WEIB(28,11.6)	28+WEIB(15,6.24)	33+EXPO(42)
59	Siesta Konutları Yolu	Esin_Sitesi	25 + ERLA(87.7, 1)	18 + WEIB(18.5, 1.17)	28+WEIB(21.5,11.17)	33+EXPO(17)	17+EXPO(23)	31+EXPO(32)
60	Siesta Konutları	Esin_Sitesi	26 + EXPO(29.3)	NORM(29.6, 18.6)	NORM(32.6,21.6)	11+ EXPO(24.3)	20+EXPO(24)	23+EXPO(32)
61	İzsu	Esin_Sitesi	30 + EXPO(27.6)	14 + EXPO(18.7)	19+ EXPO(23.7)	15+ EXPO(24.6)	18+EXPO(18)	32+EXPO(29)
62	Bahçelievler	Bahçelievler	26 + EXPO(19.6)	23 + WEIB(14.2, 0.436)	31+WEIB(18.2,5.436)	10+ EXPO(12.6)	13+EXPO(20)	33+EXPO(21)
63	Telekom	Bahçelievler	30 + EXPO(37.8)	15 + EXPO(19.4)	19+ EXPO(44)	14+ EXPO(31.8)	24+EXPO(30)	21 + WEIB(40.2, 0.625)
64	Okul_Bahçelievler	Bahçelievler	26 + WEIB(6.47, 0.563)	13 + EXPO(15.1)	16+ EXPO(20.1)	13+LOGN(17.1,6.7)	15 + WEIB(14.2, 0.664)	31+EXPO(21)

APPENDIX - 34: Statistical Distributions for passenger arrivals (cont.)

Bus Stop #	Bus Stop Name	Route	Weekdays Model Slot1	Weekdays Model Slot2	Weekdays Model Slot3	Weekend Model Slot1	Weekend Model Slot2	Weekend Model Slot3
65	Bahriye Üçok	Bahçelievler	UNIF(26, 90)	15 + EXPO(13.5)	20+ EXPO(16.5)	16+ EXPO(18.5)	15+EXPO(16)	32+EXPO(26)
66	Cumhuriyet Mah. Muhtarlığı	Harmandalı	26 + EXPO(9.04)	18 + EXPO(14.9)	24+ EXPO(18.9)	11+ EXPO(32)	25+EXPO(14)	20+EXPO(40)
67	Harmandalı Mezarlık	Harmandalı	16 + EXPO(14.8)	27 + WEIB(25.2, 0.766)	26 + EXPO(62.1)	2+ EXPO(8.8)	17+EXPO(19)	21 + WEIB(43.7, 0.517)
68	Harmandalo İÖO	Harmandalı	26 + EXPO(10.8)	18 + EXPO(18.5)	24+ EXPO(23.5)	11+ EXPO(27)	10+EXPO(31)	29+EXPO(41.4)
69	Sağlık Ocağı	Harmandalı	26 + EXPO(17.8)	23 + EXPO(11.3)	26+ EXPO(13.3)	17+ EXPO(13.8)	10+EXPO(19)	29+EXPO(23.2)
70	Zambak Durağı	Harmandalı	26 + EXPO(29.9)	18 + EXPO(16.1)	22+ EXPO(20.1)	10+ EXPO(24.9)	23+EXPO(23)	32+EXPO(32.3)
71	Papatya Durağı	Harmandalı	26 + EXPO(11.8)	23 + EXPO(27.7)	25+ EXPO(32.7)	15+ EXPO(33)	25+EXPO(27)	19 + EXPO(80.5)
72	Eski Pazar Yeri	Harmandalı	26 + EXPO(16)	30 + EXPO(5.1)	35+ EXPO(10.1)	11+ EXPO(37)	17+EXPO(27)	30+EXPO(29.5)
73	Meydan yolu-Jandarma	Harmandalı	26 + WEIB(26.6, 0.533)	14 + EXPO(18.1)	19+ EXPO(20.1)	24 + 68 * BETA(2.67, 8.52)	15 + LOGN(24.6, 45.1)	20+EXPO(19.5)
74	Doyuran Pide	Harmandalı	26 + EXPO(28.5)	20 + EXPO(24.4)	25+ EXPO(30.4)	15+ EXPO(22.5)	23+EXPO(15)	27+EXPO(18.6)
75	Eski Sinema Durağı	Harmandalı	26 + EXPO(31.2)	15 + EXPO(22.1)	17+ EXPO(27.1)	12+ EXPO(24.2)	16+EXPO(26)	21 + WEIB(49.1, 0.523)
76	Balatçık Kahveler	Harmandalı	26 + EXPO(5.36)	21 + WEIB(54.2, 0.757)	35 + WEIB(33.8, 0.398)	13+ EXPO(36)	16+EXPO(17)	UNIF(21, 210)
77	Cami Durağı	Harmandalı	26 + EXPO(7.47)	14 + EXPO(15)	19+ EXPO(18)	16+ EXPO(5.47)	16+EXPO(26)	33+EXPO(20.2)
78	Çiğli Aktarma	Harmandalı	NORM(48.2, 14.2)	15 + EXPO(15.7)	19+ EXPO(18.7)	NORM(48.2, 14.2)	15 + GAMM(37.8, 0.712)	32+EXPO(31.3)

APPENDIX - 34: Statistical Distributions for passenger arrivals (cont.)

Bus Stop #	Bus Stop Name	Route	Weekdays Model Slot1	Weekdays Model Slot2	Weekdays Model Slot3	Weekend Model Slot1	Weekend Model Slot2	Weekend Model Slot3
78	Çiğli Aktarma	Harmandalı	NORM(48.2, 14.2)	15 + EXPO(15.7)	19+ EXPO(18.7)	NORM(48.2, 14.2)	15 + GAMM(37.8, 0.712)	32+EXPO(31.3)
79	İstasyon altı-2	Harmandalı	26 + ERLA(8.39, 1)	16 + EXPO(25.1)	18+ EXPO(28.1)	28+WEIB(24.2,8.4)	15 + WEIB(15.4, 0.617)	31+EXPO(36.4)
80	Elit.Sit. Otobüs Son Durak	Elit_Sitesi	26 + GAMM(13.1, 1.47)	15 + EXPO(26.7)	18+ EXPO(31.7)	16+ EXPO(18.5)	10+EXPO(20)	20+EXPO(33)
81	Halkmar	Elit_Sitesi	NORM(40, 9.87)	13 + EXPO(19.9)	15+ EXPO(23.9)	NORM(40, 9.87)	15 + LOGN(11.8, 20.3)	UNIF(21, 210)
82	Martı	Elit_Sitesi	26 + WEIB(0.2, 0.227)	20 + EXPO(14.2)	25+ EXPO(17.2)	16+ EXPO(18.5)	13+EXPO(23)	30+EXPO(18)
83	H.Ali Gültekin Cami	Elit_Sitesi	30 + WEIB(8.4, 2.36)	12.5 + EXPO(12.3)	26 + WEIB(21.5, 0.362)	16+ EXPO(18.5)	23+EXPO(18)	21+EXPO(28.2)
84	Anadolu Eczanesi	Elit_Sitesi	26 + EXPO(10.4)	12.5 + ERLA(6.99, 2)	35 + EXPO(30)	16+ EXPO(32)	11+EXPO(20)	23 + WEIB(62.5, 0.685)
85	Toki Lisesi	Elit_Sitesi	26 + WEIB(0.13, 0.207)	12.5 + WEIB(13.2, 1.03)	NORM(78.8, 15.2)	16+ EXPO(32)	13+EXPO(13)	33+EXPO(22.2)
86	7488/4 Sokak	Elit_Sitesi	26 + EXPO(26)	12.5 + EXPO(14)	26 + EXPO(57.8)	NORM(41, 11.2)	15 + 135 * BETA(0.653, 2.3)	23+EXPO(27.8)
87	Ata Durağı_Elit Sit.	Elit_Sitesi	26 + WEIB(21.1, 1.35)	12.5 + LOGN(16, 6.4)	30 + EXPO(68.1)	14 + LOGN(12.5, 6.1)	15 + EXPO(20.2)	26 + EXPO(85.6)
88	Park	Elit_Sitesi	26 + EXPO(18)	.5 + 79 * BETA(0.207, 1.28)	79 * BETA(5.2,5.28)	19+ EXPO(33)	20+EXPO(24)	30+EXPO(28.9)
89	Yesugay Parkı	Elit_Sitesi	26 + EXPO(15)	22.5 + EXPO(9.96)	26 + EXPO(66.5)	14+ EXPO(42)	10+EXPO(33)	32+EXPO(33)
90	Eminay Özmeriç İÖO	Elit_Sitesi	26 + WEIB(0.994, 0.281)	14 + EXPO(20.5)	20+ EXPO(25.5)	16+ EXPO(18.5)	20+EXPO(27)	24+EXPO(30)
91	7400 Sokak	Elit_Sitesi	26 + EXPO(20.7)	20 + EXPO(36.6)	23+ EXPO(42.6)	16+ EXPO(16.7)	25+EXPO(16)	22+EXPO(34.4)

APPENDIX - 34: Statistical Distributions for passenger arrivals (cont.)

Bus Stop #	Bus Stop Name	Route	Weekdays Model Slot1	Weekdays Model Slot2	Weekdays Model Slot3	Weekend Model Slot1	Weekend Model Slot2	Weekend Model Slot3
91	7400 Sokak	Elit_Sitesi	26 + EXPO(20.7)	20 + EXPO(36.6)	23+ EXPO(42.6)	16+ EXPO(16.7)	25+EXPO(16)	22+EXPO(34.4)
92	Karaali Caddesi	Elit_Sitesi	26 + EXPO(5.17)	23 + EXPO(28.4)	26+ EXPO(32.4)	14+ EXPO(15)	23+EXPO(30)	19 + WEIB(71.9, 0.689)
93	Şemikler Yeni Bulvar	Şemikler	26 + WEIB(1.95, 0.293)	23 + EXPO(34.7)	28+ EXPO(39.7)	31 + 78 * BETA(3.12, 6.82)	15 + EXPO(31.7)	32+EXPO(39.2)
94	Diş Hastanesi	Şemikler	26 + EXPO(14.3)	38 + EXPO(24.3)	44+ EXPO(28.3)	11+ EXPO(11.3)	15+EXPO(29)	19 + EXPO(48.4)
95	Kipa Ekspres	Şemikler	26 + EXPO(8.1)	23 + EXPO(28.3)	28+ EXPO(32.3)	11+ EXPO(5.1)	11+EXPO(28)	22+EXPO(33.3)
96	Alt Geçit	Şemikler	25 + EXPO(47.7)	19.5 + EXPO(20.3)	30 + EXPO(58.4)	16+ EXPO(18.5)	14+EXPO(19)	42 + EXPO(60.9)
97	Cihat Kora And.Lisesi	Şemikler	26 + EXPO(24.1)	19.5 + EXPO(38)	26 + WEIB(22.5, 0.316)	12+ EXPO(19.1)	10+EXPO(14)	31+EXPO(20)
98	Başkent Hastahanesi	Şemikler	26 + EXPO(34)	21 + EXPO(41.3)	25+ EXPO(46.3)	13+ EXPO(29)	22+EXPO(26)	30+EXPO(23)
99	Top Sahası	Şemikler	26 + WEIB(0.212, 0.233)	27 + EXPO(28.3)	31+ EXPO(32.3)	NORM(52, 11.2)	15 + LOGN(35.7, 72.1)	26+EXPO(41.4)
100	Hamdullah Suphi İÖO	Şemikler	26 + EXPO(6.86)	43 + EXPO(25.8)	49+ EXPO(29.8)	15+ EXPO(38)	24+EXPO(30)	23 + WEIB(54.9, 0.631)
101	Yalı Çocuk Parkı	Şemikler	26 + EXPO(23.8)	50 + WEIB(9.43, 0.742)	14+WEIB(20.43,3.742)	18+ EXPO(19.8)	12+EXPO(27)	32+EXPO(43)
102	Blue City Hotel	Şemikler	30 + WEIB(1.5, 0.212)	50 + EXPO(14.9)	55+ EXPO(16.9)	16+ EXPO(18.5)	10+EXPO(24)	UNIF(19, 210)
103	Kapalı Spor Salonu	Şemikler	26 + WEIB(0.985, 0.258)	50 + WEIB(7.49, 0.6)	20+WEIB(17.49,9.6)	31+EXPO(21)	13+EXPO(31)	22+EXPO(35)
104	Carrefour Durağı	Şemikler	26 + WEIB(2.9, 0.261)	50 + EXPO(11.3)	54+ EXPO(14.3)	27+EXPO(22)	18+EXPO(20)	30+EXPO(24)
105	Bozoğlu Konutları	Şemikler	26 + EXPO(12)	50 + EXPO(14.2)	55+ EXPO(16.2)	9+ EXPO(35)	11+EXPO(27)	15 + GAMM(32.5, 1.13)
106	Pazaryeri_Elit Sit	Elit_Sitesi	26 + EXPO(19.9)	50 + 70 * BETA(0.438, 0.771)	70 * BETA(10.43,9.771)	13+ EXPO(15.9)	19+EXPO(17)	26+EXPO(18)

APPENDIX - 34: Statistical Distributions for passenger arrivals (cont.)

Bus Stop #	Bus Stop Name	Route	Weekdays Model Slot1	Weekdays Model Slot2	Weekdays Model Slot3	Weekend Model Slot1	Weekend Model Slot2	Weekend Model Slot3
107	Tansaş	Evka-2	26 + EXPO(15.9)	20 + EXPO(25)	26+ EXPO(27)	16+ EXPO(9.9)	13+EXPO(15)	26+EXPO(38)
108	Tansaş_Elit Sitesi	Elit_Sitesi	26 + EXPO(27)	25 + EXPO(38.4)	28+ EXPO(40.4)	13+ EXPO(20)	21+EXPO(22)	30 + GAMM(97.4, 0.552)
109	Anadolu Caddesi	Egekent	15 + EXPO(11.4)	7.5 + EXPO(7.95)	25.5 + EXPO(5.64)	5+ EXPO(22)	10+EXPO(17)	19 + EXPO(29.3)
110	Ata Durağı	Evka-5	10.5 + 29.5 * BETA(0.932, 2.39)	15 + EXPO(12.2)	19+ EXPO(17.2)	NORM(34, 12)	15 + WEIB(10, 0.509)	23+EXPO(30)
111	Atakent İÖO	Tersane	10.5 + WEIB(10.9, 1.04)	20 + EXPO(8.48)	23+ EXPO(13.48)	29+EXPO(24)	18+EXPO(25)	19 + WEIB(35.6, 0.674)
112	Atatürk Bulvarı-Garanti Bankası	Tersane	11.5 + EXPO(17.1)	25 + WEIB(9.58, 1.16)	37+WEIB(13.58,4.16)	11+ EXPO(12.1)	10+EXPO(13)	32+EXPO(30.1)
113	Barış Durağı	Tersane	11 + EXPO(15.5)	11.5 + EXPO(11.9)	25.5 + EXPO(13.9)	11+ EXPO(13.5)	20+EXPO(26)	23+EXPO(22)
114	Beşikçioğlu Camii	Tersane	20 + WEIB(10.8, 1.36)	7.5 + LOGN(14.7, 41.6)	26 + EXPO(19)	26+EXPO(55)	15+EXPO(37)	27+EXPO(46)
115	Cami	Evka-5	20 + WEIB(30.1, 1.25)	40 + WEIB(7.02, 1.9)	46+WEIB(10.02,8.9)	31+EXPO(22)	25+EXPO(29)	22+EXPO(40)
116	Çiğli Kipa Son Durak	Tüm_Otobüsler	28 + WEIB(0.625, 0.150)	20 + EXPO(53)	26+EXPO(55)	24+EXPO(18)	18+EXPO(28)	TRIA(19, 38.1, 210)
117	Dört Yol	Bostanlı	26 + WEIB(0.784, 0.257)	20 + EXPO(34.4)	22+ EXPO(38.4)	30+EXPO(25)	25+EXPO(18)	31+EXPO(22)
118	Emniyet	Tersane	15 + WEIB(17, 1.13)	24 + EXPO(28.1)	30+EXPO(31.1)	26+EXPO(22)	11+EXPO(13)	20+EXPO(33)
119	Eski son durak	Evka-5	13 + WEIB(7.37, 0.342)	22 + WEIB(8.66, 1.94)	18+ WEIB(12.66,5.94)	30+EXPO(23)	12+EXPO(21)	19 + WEIB(30, 0.64)
120	Geçit-1	Tersane	12 + ERLA(14.3, 1)	UNIF(20, 45)	35 + EXPO(28.5)	31+EXPO(23)	17+EXPO(15)	28+EXPO(39)
121	Geçit-2	Tersane	20 + GAMM(9.01, 2)	15 + EXPO(13.8)	20+ EXPO(18.8)	22+EXPO(26)	22+EXPO(20)	22+EXPO(35)
122	Göz Hastanesi	Tersane	30 + WEIB(11.2, 2.44)	15 + EXPO(7.58)	18+ EXPO(13.58)	33+EXPO(29)	18+EXPO(26)	33+EXPO(28)

APPENDIX - 34: Statistical Distributions for passenger arrivals (cont.)

Bus Stop #	Bus Stop Name	Route	Weekdays Model Slot1	Weekdays Model Slot2	Weekdays Model Slot3	Weekend Model Slot1	Weekend Model Slot2	Weekend Model Slot3
122	Göz Hastanesi	Tersane	30 + WEIB(11.2, 2.44)	15 + EXPO(7.58)	18+ EXPO(13.58)	33+EXPO(29)	18+EXPO(26)	33+EXPO(28)
123	Koçtaş	Tersane	7 + WEIB(1.26, 0.329)	8.5 + EXPO(8.38)	19 + EXPO(11.8)	41+WEIB(18,8.9)	15 + 135 * BETA(1.03, 12.2)	20+EXPO(24)
124	Lunapark	Tersane	7 + WEIB(15.7, 1.54)	UNIF(15.5, 25)	26 + EXPO(36.9)	33+EXPO(22)	17+EXPO(16)	19 + WEIB(11.6, 0.406)
125	Mavibahçe Durağı	Tersane	15 + EXPO(21.2)	8.5 + EXPO(4.67)	30 + EXPO(28.1)	8+ EXPO(19.2)	21+EXPO(27)	25+EXPO(29)
126	Okul	Evka-5	12 + ERLA(10.8, 1)	13.5 + EXPO(8)	26 + EXPO(31.2)	21 + ERLA(14.3, 1)	15 + EXPO(17.7)	29+EXPO(36.1)
127	Migros Durağı	Tersane	11 + EXPO(8.16)	13.5 + GAMM(4.15, 1.7)	26 + EXPO(40)	4+ EXPO(2.16)	20+EXPO(18)	18.5 + GAMM(20.7, 1.18)
128	Otoban	Yeni_Girne	22 + EXPO(18)	5.5 + ERLA(5.21, 1)	19 + EXPO(12.1)	5+ EXPO(11)	12+EXPO(26)	21 + WEIB(33.2, 0.643)
129	Pazaryeri	Tersane	TRIA(5, 12, 50)	14 + WEIB(7.27, 0.624)	18+ WEIB(14.27,5.624)	26+EXPO(15)	14+EXPO(22)	27+EXPO(28)
130	Santral	Evka-5	30 + WEIB(12.2, 0.984)	14 + EXPO(9.52)	19+ EXPO(14.52)	28+EXPO(19)	13+EXPO(18)	27+EXPO(24)
131	Şaraphane	Evka-5	12 + EXPO(18.9)	14 + EXPO(11)	20+ EXPO(14)	8+ EXPO(13.9)	16+EXPO(23)	24+EXPO(27)
132	Şehit Üst. ONUR ÖZKAYA	Evka-2	10.5 + EXPO(11.3)	12.5 + EXPO(15.6)	30 + EXPO(28.4)	6+ EXPO(18)	25+EXPO(19)	26 + WEIB(42.8, 0.656)
133	Yıllar	Tersane	11 + EXPO(9.44)	12.5 + EXPO(6.73)	26 + EXPO(33.1)	4+ EXPO(34.4)	20+EXPO(24)	21+EXPO(33)
134	Yunuslar	Tersane	13.5 + EXPO(13.8)	11.5 + EXPO(8.85)	26 + EXPO(34.7)	7+ EXPO(6.8)	18+EXPO(16)	21 + WEIB(72.2, 0.672)
135	Dört Yol Bahçelievler	Bahçelievler	26 + ERLA(14.7, 1)	20 + EXPO(34.2)	24+ EXPO(36.2)	24+EXPO(16)	13+EXPO(16)	21+EXPO(40.9)

APPENDIX - 35: Discrete distribution for direction of passengers

#	Route	Discrete Function
1	Evka-5	DISC(0.26,21,0.47,22,0.56,23,0.69,130,0.78,24,0.8,25,0.81,26,0.82,27,0.86,28,0.94,29,0.96,131,0.97,126,0.98,127,0.99,133,1,134)
2	Evka-2	DISC(0.31,30,0.33,31,0.37,32,0.42,33,0.51,34,0.57,35,0.65,123,0.71,36,0.73,37,0.74,38,0.75,39,0.76,40,0.84,41,0.91,42,0.92,43,1,135)
3	Egekent	DISC(0.09,44,0.28,130,0.4,45,0.41,46,0.52,127,0.64,47,0.72,48,0.77,49,0.78,50,0.82,131,0.84,51,0.86,52,0.88,53,0.95,54,0.96,55,0.99,125,1,132)
4	Tersane	DISC(0.21,57,0.31,58,0.39,59,0.45,6,0.57,7,0.72,8,0.74,9,0.76,10,0.78,11,0.82,12,0.83,13,0.84,14,0.9,15,0.91,16,0.95,17,0.97,18,0.98,19,1,20)
5	Yeni_Girne	DISC(0.25,60,0.45,61,0.58,62,0.7,63,0.79,64,0.82,65,0.83,66,0.84,67,0.89,68,0.92,69,0.96,70,0.97,136,0.98,132,1,137)
6	Bostanlı	DISC(0.39,71,0.44,14,0.66,15,0.78,16,0.89,72,0.93,129,0.95,73,0.96,74,0.98,19,1,20)
7	Esin_Sitesi	DISC(0.1,75,0.31,76,0.34,77,0.38,135,0.53,78,0.71,79,0.9,80,0.98,81,1,132)
8	Bahçelievler	DISC(0.21,57,0.31,58,0.39,59,0.45,6,0.57,7,0.72,8,0.74,9,0.76,10,0.78,11,0.82,12,0.83,13,0.84,14,0.9,15,0.91,16,0.95,17,0.97,18,0.98,19,1,20)
9	Harmandalı	DISC(0.2,82,0.36,83,0.5,84,0.59,85,0.69,86,0.78,87,0.82,88,0.84,89,0.86,90,0.88,125,0.91,91,0.93,92,0.95,126,0.96,93,0.97,133,0.98,134,0.99,94,1,95)
10	Elit_Sitesi	DISC(0.25,96,0.36,97,0.54,124,0.61,98,0.68,99,0.77,100,0.84,101,0.86,102,0.88,122,0.9,103,0.93,104,0.95,105,0.97,106,0.98,107,0.99,108,1,132)

APPENDIX - 40: SIMAN view for passenger arrival

```
0$          CREATE,
1,60:ED(TimeSlot+1+(1*3)),
Stop_Creating_Entities:MARK(Timein):NEXT(57$);

57$          BRANCH,          1:

If,Stop_Creating_Entities == 0,58$,Yes:
Else,55$,Yes;
58$          DISPOSE:          No;

55$          ASSIGN:          Driver=2;
Entity Type 1 = Driver

56$          STATION,
Bahcelievler_Bahcelievler;
25$          BRANCH,          1:
If,Driver==1,27$,Yes:
Else,26$,Yes;

Entity
26$          QUEUE,
Bahcelievler_Q:DETACH;
```

APPENDIX - 41: SIMAN view for movement between stations

```

0$          CREATE,          1,60:ED(TimeSlot+1+(1*3)),
Stop_Creating_Entities:MARK(Timein):NEXT(57$);

57$          BRANCH,          1:
          If,Stop_Creating_Entities
== 0,58$
,Yes:
          Else,55$,Yes;

58$          DISPOSE:          No;

55$          ASSIGN:          Driver=2;
Entity Type 1 = Driver

56$          STATION,          Bahcelievler_Bahcelievler;
25$          BRANCH,          1:
          If,Driver==1,27$,Yes:
          Else,26$,Yes;

Entity Type Query
27$          BRANCH,          1:
          If,Direction==1,29$,Yes:
          Else,51$,Yes;

Direction 1 = To the mall
29$          BRANCH,          1:

          If,NQ(Bahcelievler_Q)>0,30$,Yes:
          Else,35$,Yes;

Is there any waiting passengers in the queue ?
30$          BRANCH,          1:

          If,Bahcelievler_Available_Seats>0,
28$,Yes:
          Else,33$,Yes;

Is there any empty seat in the bus ?
28$          DELAY:          0.087 + 0.094 *
(MN(Bahcelievler_Available_Seats,NQ(Bahcelievler_Q)))
,,Other:NEXT(31$); Boarding Time

31$          PICKUP:
Bahcelievler_Q,1,MN(Bahcelievler_Available_Seats,
NQ(Bahcelievler_Q)); Pickup passengers

```

APPENDIX - 41: SIMAN view for movement between stations (cont)

```
34$          ASSIGN:          IS=1:

Bahcelievler_Available_Seats=
EmptyBusSize-NG;          Update number of
available seats
32$          TRANSPORT:
Bahcelievler_35KE8384,SEQ,UNIF(
Bahcelievler_Min_Velocity,
Bahcelievler_Max_Velocity); Go to the next station

33$          TRANSPORT:
Bahcelievler_35KE8384,Kipa_Giris,UNIF(
Bahcelievler_Min_Velocity,
Bahcelievler_Max_Velocity);

35$          TRANSPORT:
Bahcelievler_35KE8384,SEQ,UNIF(
Bahcelievler_Min_Velocity,
Bahcelievler_Max_Velocity); Go to the next station

51$          BRANCH,          1:

If,Bahcelievler_Available_Seats==27
,39$,Yes:

Else,50$,Yes;

Is there any passenger in the bus
at the last station in
the sequence
39$          ASSIGN:
Bahcelievler_Available_Seats=27:
Direction=1:

Bahcelievler_35KE8384_Travel_Number=
Bahcelievler_35KE8384_Travel_Number+1:
NS=1:
IS=1;

52$          BRANCH,          1:

If,Bahcelievler_35KE8384_Travel_Number == 19,53$,Yes:
Else,40$,Yes;

53$          FREE:          Bahcelievler_35KE8384;
54$          DISPOSE:          No;
```

APPENDIX - 42: SIMAN view of experimental frame

```

PROJECT,      "Weekdays Model", "Dogan Can
GOKMEN", 06/12/2020, Yes, No, No, No, No, No, No, No, No, No, No, No, No;

ATTRIBUTES:   Attributes of model

VARIABLES:    Variables of model

QUEUES:       Queues of model

STATIONS:     Bus stops of model

DISTANCES:    Distances between bus stops

SEQUENCES:    Routes for buses

TALLIES:      1, Passengers_ArrivedTimeIn:
              2, Passengers_DroppedoffTimeIn;

OUTPUTS:      1, TNUM(Passengers_ArrivedTimeIn), "passengerscoming.dat":
              2, TNUM(Passengers_DroppedoffTimeIn), "passengersgoing.dat":

              3, (((TNUM(Passengers_ArrivedTimeIn)+TNUM(Passengers_DroppedoffTimeIn)) /
              (384*27))) / NREP) * 100, "Fulness_Ratio.dat",
              Fulness_Ratio:

              4, TNUM(Passengers_ArrivedTimeIn)+TNUM(Passengers_DroppedoffTimeIn), "totaln
              umber.dat", Total_Passengers_Number:
              5,

              ((TAVG(Passengers_ArrivedTimeIn)*TNUM(Passengers_ArrivedTimeIn))+(TAVG(Pas
              sengers_DroppedoffTimeIn)*TNUM(Passengers_DroppedoffTimeIn)))/(TNUM(Passen
              gers_DroppedoffTimeIn)+TNUM(Passengers_ArrivedTimeIn)),
              "ATT.dat", Average_Travel_Time;

REPLICATE,    20, 0.0, , Yes, No, 0.0, , , 24.0, Minutes, No, No, , DATETIME("Dec 06,
2020 15:53:21"), Yes;

EXPRESSIONS:  Statistical Distributions for passenger arrivals

```

APPENDIX - 43: SIMAN view of time slot chancing

```
2266$          CREATE,  
1,0:,1:NEXT(2269$);  
  
2269$          ASSIGN:  
TimeSlot=-3:  
SetVelocity:  
  
2268$          DELAY:  
180,,Other:NEXT(2271$);  
  
2271$          ASSIGN:  
TimeSlot=-2:  
  
Week_Days=0:  
SetVelocity:  
  
2270$          DELAY:  
300,,Other:NEXT(2272$);  
  
2272$          ASSIGN:  
TimeSlot=-1:  
  
Week_Days=0:  
SetVelocity:  
  
2267$          DISPOSE:  
No;
```

APPENDIX - 44: SIMAN view for passenger pick up logic

Entity	Type	Query
2299\$	BRANCH,	1:
If, Bus_Plaque==Evka5_1_35KE8135, 2300\$, Yes:		
If, Bus_Plaque ==		
Evka5_2_35AED896, 2328\$, Yes;		
2300\$	BRANCH,	1:
If, Direction==1, 2302\$, Yes:		
Else, 2324\$, Yes;		
Direction 1 = To the mall		
2302\$	BRANCH,	1:
If, NQ(Evka5_Toki_Q)>0, 2303\$, Yes:		
Else, 2308\$, Yes;		
Is there any waiting passengers in the queue ?		
2303\$	BRANCH,	1:
If, Evka5_1_35KE8135_AvailableSeats > 0, 2301\$, Yes:		
Else, 2306\$, Yes;		
Is there any empty seat in the bus ?		
2301\$	DELAY:	0.087 + 0.094 *
(MN(Evka5_1_35KE8135_AvailableSeats, NQ(Evka5_Toki_Q))), ,		
Other: NEXT(2304\$); Boarding Time		
2304\$	PICKUP:	
Evka5_Toki_Q, 1, MN(Evka5_1_35KE8135_AvailableSeats, NQ(Evka5_Toki_Q)); Pickup passengers		
2307\$	ASSIGN:	IS=1:
Evka5_1_35KE8135_AvailableSeats=EmptyBusSize-NG;		
Update number of available seats		

APPENDIX - 45: SIMAN view for passenger drop off logic

```
2305$          TRANSPORT:
Evka5_1_35KE8135,SEQ,UNIF(Evka5_Min_Velocity,Evka5_Max_V
elocity); Go to the next station

2306$          TRANSPORT:
Evka5_1_35KE8135,Kipa_Giris,UNIF(Evka5_Min_Velocity,Evka
5_Max_Velocity);

2308$          TRANSPORT:
Evka5_1_35KE8135,SEQ,UNIF(Evka5_Min_Velocity,Evka5_Max_V
elocity); Go to the next station

2324$          BRANCH,          1:
If,Evka5_1_35KE8135_AvailableSeats==27,2312$,Yes:
Else,2323$,Yes;                                Is
there any passenger in the bus at the last station in
the sequence

2323$          DELAY:          0.087 +( 0.094 *
NG),,Other:NEXT(2309$);          Boarding Time

2309$          DROPOFF,          1,NG:2310$:NEXT(2312$);

2310$          TALLY:
Passengers_DroppedoffTimein,INT(Timein),1;
2311$          DISPOSE:          No;
```