DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

SOLVING TRAIN SCHEDULING PROBLEM BY USING SIMULATION OPTIMIZATION

by

Mustafa AKDEMİR

September, 2023

İZMİR

SOLVING TRAIN SCHEDULING PROBLEM BY USING SIMULATION OPTIMIZATION

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

by

Mustafa AKDEMİR

September, 2023

İZMİR

THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **“SOLVING TRAIN SCHEDULING PROBLEM BY USING SIMULATION OPTIMIZATION”** completed by **MUSTAFA AKDEMİR** under supervision of **PROF.DR. GÖKALP YILDIZ** 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.

Prof.Dr. Gökalp YILDIZ

Supervisor

Prof. Dr. Özcan KILINÇÇI

Jury Member

Dr. Öğr. Üyesi Özgür ESKİ

Jury Member

Prof. Dr. Okan FISTIKOĞLU

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENT

I would like to express my sincerest appreciation to my supervisor, Prof. Dr. Gökalp YILDIZ, for her continuous guidance, unwavering support, motivation, and invaluable counsel during the course of this thesis. Undoubtedly, his dedicated contributions have played a pivotal role in helping me achieve my goals.

I extend my heartfelt thanks to Dr. Özgür YALÇINKAYA, with whom I had the chance to start and continue this thesis together, even though we couldn't finish it together. I appreciate his collaboration and the knowledge I gained from him.

Lastly, I want to convey my special gratitude and profound thanks to my family for supporting me all the time.

Mustafa AKDEMİR

SOLVING TRAIN SCHEDULING PROBLEM BY USING SIMULATION OPTIMIZATION

ABSTRACT

Train scheduling is an important part of railway management which aims to determine a feasible timetable for a set of trains while considering track capacities and operational constraints. Train scheduling is a highly complex problem due to reasons such as a large number of variables and constraints, a vast solution space, conflicting objectives, limited resources, safety protocols, and the constant need for maintenance.

In this thesis, the existing literature on train scheduling problems is reviewed and the approaches used to tackle these challenges are examined. To address these challenges, a simulation optimization framework is considered. This approach allows us to account for the inherent stochasticity of railway systems. In this thesis, the aim is to build upon the proposed problem and simulation model by Yalçınkaya (2010) to improve the results and identify the most suitable metaheuristic algorithm. A comprehensive analysis of various metaheuristic algorithms is conducted, including Genetic Algorithm (GA), Simulated Annealing (SA), and Tabu Search (TS), to determine their effectiveness in solving train scheduling problems. Hybridization strategies are also explored, by combining these algorithms to harness their respective strengths and improve solution quality. The findings demonstrate that integrating simulation with metaheuristics offers a promising avenue for optimizing train schedules, resulting in more efficient rail systems.

In conclusion, this thesis sheds light on the critical importance of efficient train scheduling in contemporary rail systems and presents an approach that combines simulation and metaheuristic algorithms to address the inherent complexities of the train scheduling problem. The research paves the way for further investigations into optimization techniques and their practical applications in real-world rail networks.

Keywords: Train Scheduling, Hybrid Metaheuristics, Simulation Optimization

TREN ÇİZELGELEME PROBLEMİNİN BENZETİM OPTİMİZASYONU YÖNTEMİYLE ÇÖZÜLMESİ

ÖZ

Tren çizelgeleme problemi, demiryolu yönetiminin önemli bir parçası olup, trenlerin takviye kapasitelerini ve operasyonel kısıtlamaları dikkate alarak bir dizi tren için uygun bir zaman çizelgesi belirlemeyi amaçlar. Bu tez, tren çizelgeleme problemi konusundaki karmaşık alanı inceleyerek ortalama tren seyahat süresini en aza indirgeme odaklıdır. Tren çizelgeleme problem büyük sayıda değişken ve kısıt, geniş bir çözüm kümesi, çatışan amaçlar, sınırlı kaynaklar, katı güvenlik protokolleri ve sürekli bakım ihtiyacı gibi nedenlerden dolayı son derece karmaşık bir problemdir.

Bu tezde, tren çizelgeleme problemleri üzerine mevcut literatür incelenmiş ve bu zorlukların üstesinden gelmek için kullanılan yöntemler gözden geçirilmiştir. Bu zorlukları ele almak için bir simülasyon optimizasyon çerçevesi düşünülmüştür. Bu yaklaşım, demiryolu sistemlerinin doğasındaki stokastisiteyi hesaba katmamıza olanak tanır. Bu tezde, Yalçınkaya (2010) tarafından önerilen problem ve simülasyon modeli üzerinde benzetim optimizasyonu uygulanarak sonuçları iyileştirmeyi ve en uygun metasezgisel algoritmayı belirlenmesi amaçlamaktadır. Genetik Algoritma, Benzetilmiş Tavlama ve Tabu Arama dahil olmak üzere çeşitli metasezgisel algoritmaların etkililiğini belirlemek için kapsamlı bir analiz yapılmıştır. Bu algoritmaların birleştirilmesi ile çözüm kalitesini artırmak için hibritleştirme stratejileri de incelenmiştir. Bulgular, benzetim optimizasyonunun tren çizelgeleme probleminioptimize etmek için umut verici bir yol sunduğunu göstermektedir.

Sonuç olarak, bu çalışma, çağdaş demiryolu sistemlerinde etkili tren sefer planlamanın kritik önemini ortaya koymakta ve tren sefer planlama probleminin doğasındaki karmaşıklıkları ele almak için simülasyon ve metaheuristik algoritmaların birleştirildiği bir yaklaşım sunmaktadır. Bu araştırma, gelişmiş optimizasyon tekniklerinin ve bunların gerçek dünya demiryolu ağlarındaki pratik uygulamalarının daha fazla incelenmesi için bir yol açmaktadır.

Anahtar kelimeler: Tren Çizelgeleme, Hibrit Metasezgiseller, Benzetim Optimizasyonu

CONTENTS

LIST OF FIGURES

Page

LIST OF TABLES

Page

CHAPTER ONE
INTRODUCTION

In this chapter, an introduction to the thesis is given, including its background, motivation, objectives, and the outline of the thesis structure.

1.1 Background and Motivation

The railway has consistently stood out as the most dependable, cost-effective means of transporting goods and passengers. Throughout the industrial revolution and beyond, railways have remained a vital component of a nation's economy. There are several compelling reasons driving the need for more efficient utilization and planning of rail infrastructures. One crucial factor is the ongoing transformation of railways into more liberalized and privatized entities. Additionally, by promoting the use of trains over other modes of transportation, optimized scheduling can contribute to reducing greenhouse gas emissions and environmental pollution. Furthermore, as countries seek to modernize their transportation systems and reduce dependence on road transport, the study of train scheduling aligns with global trends in sustainable mobility.

One significant challenge in railway system management pertains to the problem of scheduling trains. Train scheduling problem entails establishing a feasible timetable for a group of trains while adhering to track capacities and meeting specific operational restrictions. Various forms of this problem can be explored, primarily contingent on the optimization objective, decision-making elements, limitations, and the intricacy of the specific railroad system.

Train scheduling has attracted considerable interest from numerous researchers, leading to extensive endeavors aimed at finding solutions. The earliest scientific article addressing this problem dates back to 1966 and published by Frank (1966), and since then, numerous articles have been published on this subject. In its initial stages, restricted by the limited computational power of computers and the intricate nature of the problem, researchers often relied on simplifying assumptions, primarily exploring deterministic models. As capabilities of the computers advanced, more realistic and complex models were formulated, incorporating optimization techniques into the modeling frameworks. Researchers have been committed to creating effective solution generation techniques, with ongoing efforts in this direction.

One notable technique is simulation modeling, which provides researchers with the means to model complex problems characterized by stochastic elements. While a limited number of articles have explored the integration of simulation and metaheuristic algorithms for simulation optimization to address train scheduling problems, a comprehensive study of this implementation is lacking. This serves as the motivation for the present thesis.

1.2 Research Objectives

In this thesis, the study of the train scheduling problem is undertaken with the following objectives:

- Reviewing relevant literature.
- Developing optimization algorithms for the problem to find the most suitable one.

To achieve these objectives;

- The studies on train scheduling problem are reviewed through 2000 to 2023 and classified according to their characteristics.
- The problem and comprehensive simulation modelling framework that was developed by Yalçınkaya (2010) are used as basis for this thesis to achieve better results and compare different metaheuristic algorithms for simulation optimization.
- Simulation optimization algorithms are proposed to create a feasible schedule while optimizing the average train travel time.

1.3 Organization of the Thesis

In Chapter Two, an extensive review of the literature concerning studies on train scheduling problem is given. Chapter Three offers an overview of the hypothetical train scheduling problem (TrainSchProb) and simulation modelling framework that was proposed by Yalçınkaya (2010). Chapter Four provides a summary of the metaheuristic algorithms used in this thesis. In Chapter Five, four simulation optimization approaches are presented and tested for TrainSchProb. Lastly, in Chapter Six, final thoughts and potential avenues for future research are outlined.

CHAPTER TWO
LITERATURE REVIEW ON TRAIN SCHEDULING PROBLEM

The studies in Train Scheduling Problem can be categorized as scheduling (timetabling) and rescheduling (dispatching). Scheduling is the formation of timetable for trains by taking into consideration arrival and departure times. These studies start with an infeasible timetable with many conflicts with the aim of finding a conflict-free timetable so that trains run according to this table. Studies in second main group is involved with rescheduling trains after disturbance in system. Disturbance can occur for various reasons. Unlike scheduling, these studies usually start with a feasible timetable without conflicts. After a disruption in system, timetable must be revised. Aim of these studies is to readjust the system.

Classification of papers in this thesis composed of review, scheduling and rescheduling papers. Yalçınkaya (2010), Narayanaswami et al. (2011), Cacchiani et al. (2012), Cacchiani et al. (2014), Fang et al. (2015) and Wen et al. (2019) conducted literature reviews for this problem. Considering the content and scope of these studies, this thesis has surveyed train scheduling problems published between 2010 and 2023.

2.1 Review Papers

In the first one, Narayanaswami et al. (2011), some of the major papers about scheduling and rescheduling in railway operations are examined. A comprehensive summary of objectives and motivations for railway scheduling and rescheduling were given. Some of the methodological details were discussed. Based on solution methods and operational problems, classification of literature was presented. This review also includes some PhD theses. A few problem variations which might have research potential were reported. In another paper, Cacchiani et al. (2012) examined nominal and robust train scheduling problems. An overview and the differences of both nominal and robust models were given. After two years, Cacchiani et al. (2014) presented an overview of recovery, disruption, and disturbance management for railway operations. Microscopic and macroscopic approaches were discussed. They highlighted the importance of integration of timetable, crew scheduling and rolling stock (main rescheduling phases) for future research in this subject. In a different review paper, Fang et al. (2015) presented a survey on rescheduling problem for railway networks. A few important formulations were explained thoroughly. Various solution approaches were analyzed, and advantages and disadvantages of these approaches were debated. Problem instances and sizes were discussed. Six suggestions for future research were given. Lastly, in Wen et al. (2019), data-driven approaches on train rescheduling problems were examined. Some machine learning for train and intelligent train rescheduling approaches were suggested for future work.

2.2 Papers on Train Scheduling

These papers address the task of creating a feasible and optimal schedule for trains. Also, they contain arrival and departure times of trains. Different solution approaches had been implemented through years to find best results. In this thesis, according to their solution approaches, these papers grouped into three category which are mathematical models, simulation models and the other solution approaches.

2.2.1 Train Scheduling Using Mathematical Model

Yang et al. (2010) created a mixed integer programming model (MIP) to solve train scheduling problem. Three different criteria were proposed, and a branch-and-bound algorithm was implemented to minimize the weighted total passengers' trip time. Additionally, the model's performance was assessed on a small numerical example. In another study, Canca et al. (2011) proposed an integer non-linear programming (INLP) model in order to calculate the optimal number of trains to schedule for a given demand as well as compute the optimal scheduling for these trains. Both the quality of the service and the total average waiting time for passengers considered. The goal of this formulation is to compute arrival and departure times and derive important metrics for analysis. Liu et al. (2011) proposed a mathematical model for minimizing makespan considering prioritized and non-prioritized trains. To find an efficient train schedule, a two-stage heuristic algorithm was constructed. Experiments showed that proposed methodology is generic and could be a standard tool for train scheduling problems. Mu et al. (2011) developed two mathematical model and optimization-based heuristic solution approaches for train scheduling problem. One of these two solution approaches is an improved FlexiblePath algorithm and the other one is a mix of genetic and FixedPath algorithm. According to experiment results, these algorithms outperform two existing algorithms. Yang et al. (2012) formulated a mathematical model for maximizing overlapping time. Secondly, some scheduling rules were proposed and to solve the integer linear programming (ILP) model, a genetic algorithm (GA) was implemented. A large-scale experiment was conducted, and promising results were shown. In another study, Tang et al. (2021) explored the train scheduling problem on suburban rail transit lines. A GA is suggested for MILP and MINLP models with the objective of minimize the total passenger waiting time. A case study was conducted and showed the promising performance of the proposed algorithm. Gong et al. (2021) suggested a variable neighborhood search (VNS) for a stochastic train scheduling problem to minimize weighted total cost for both operators and passengers. Effectiveness of the proposed method was tested with two sets of numerical experiments. It was shown that VNS performs well although it has some limitations. Polinder et al. (2021) formulated an integer quadratic programming model and proposed a heuristic approach to minimize travel time. Promising results were presented by case studies. Zhou et al. (2022) tried a combined approach to optimize train scheduling, passenger flow control strategy, and rolling stock circulation planning. They formulated a MILP model and proposed a tabu search (TS) heuristic to tackle the problem. Their findings indicate that employing a variety of rolling stock types can significantly enhance the metro system, improving service quality and reducing operating costs. Zhang et al. (2022) combined the optimization of train scheduling and track maintenance scheduling. They suggested a binary integer program model and proposed a dynamic heuristic approach to minimize total weighted operating and maintenance costs. They showed that proposed approach outperforms the commercial solver. Wang et al. (2023) suggested simulation optimization approach to handle the train scheduling problem. They formulated an INLP model to both minimize passenger waiting time and the total operating cost. To generate pareto optimal results, a ε-constrain method is proposed.

Unlike many other studies, Li et al. (2013) proposed a multi-objective train scheduling model which aims to minimize both total passenger travel time and minimizing energy consumption. In order to solve proposed model, a fuzzy multi-objective solution algorithm was introduced. Compared to the existing models, this green scheduling model can reduce carbon emission greatly. Similar to this study, Hu et al. (2013) constructed a mathematical optimization model which minimizes both emission cost and total passenger trip time. A fuzzy multi-objective algorithm was proposed to solve the mathematical model. Proposed model compared with the existing models with the use of numerical experiments. Zhou et al. (2023) suggested a MILP model to minimize the total energy consumption as well as the operating cost of rolling stocks. They proposed a particle swarm optimization method and demonstrated results in a real-word case.

Espinosa et al. (2014) proposed a MILP model to maximize profit for high-speed rail systems. The model consists of both of an infrastructure model and a demand model. A real-world experiment was conducted. Li et al. (2014) suggested a dynamic train scheduling and a control framework in order to adjust cycle time, speed and timetable of trains while minimizing energy consumption. Focus of this study is both scheduling and speed of trains. Promising results were shown with real-life experiments. Dollevoet et al. (2014) suggested an optimization framework for both train scheduling and delay management. Two mathematical model for these steps were proposed. Delay management and train scheduling models were combined in a framework. Variables of delay management were used to define train scheduling problem. A real-world experiment was conducted, and results were shared. Yang et al. (2015) proposed a method that minimizes both energy consumption and travel time. An ILP model with speed and timetable was constructed. Additionally, an adaptive GA and control algorithm were presented. Meaningful improvements were gained. Qi et al. (2016) integrated a single-level and a bi-level mathematical optimization model. One is for train scheduling and the other one is for platform choice. A suitable local search algorithm was used to find good results. Cacchiani et al. (2016) suggested an ILP model to solve the train scheduling problem. Highly congested railways were the primary focus for study. In order to get dual bounds, linear relaxation method was used. In addition, a heuristic algorithm was proposed. Algorithm was proved to be effective even for very large instances. Huang et al. (2016) formulated a multi-objective model to minimize energy consumption and maximize service quality. In order to enhance the solution's quality, a GA with binary encoding is proposed. Passenger travel time, energy consumption, service level and operating cost are the main focuses of this study. Yang et al. (2016) formulated a multi-objective MILP model for minimizing the total dwelling time and total delay for high-speed railway. A small example and a real-life experiment were demonstrated, and model showed promising results. Yin et al. (2017) formulated an ILP model and a MILP model for timetabling. One is for minimizing passenger waiting time and other one is for maximizing utilization of regenerative braking energy. To overcome the complexity of these models, a lagrangian relaxation-based heuristic was implemented. Lamorgese et al. (2017) created a MILP model for time tabling. Bender's decomposition method was used to find optimal results. Model finds optimal solutions in a reasonably short time. Huang et al. (2017) proposed a multi-objective mathematical model with the objectives of energy efficiency and high service quality. Three types of energy consumption were considered in the model. A TS algorithm was used to solve mathematical model. Corman et al. (2017) formulated and integrated a MILP model for delay management and timetabling with the aim of minimizing travel time. Four tailored heuristic algorithms were implemented to find good results in a reasonable time. Model was applied to real-life applications. Shi et al. (2018) constructed an ILP model to minimizing passenger waiting time. Local search heuristic was used to solve the model. Liu et al. (2018) created a mathematical model for energy-efficient train timetabling. A dynamic programming approach was used to solve the model. Results were compared to other solution approaches. Yu et al. (2019) suggestted a MIP model which deals with both passenger path assignment and last train scheduling. The proposed model can be solved with the standard solvers without the need of using any heuristic approaches to find optimal solution. Wang et al. (2019) suggested a MILP model for scheduling last and first trains with the aim of minimizing total travel time and passenger waiting time. In order to solve model, a Lagrangian relaxation-based decomposition method was used. Garrisi et al. (2020) formulated a mixed integer linear programming linear programming (MILP) model for train scheduling. For solving the model, a genetic heuristic algorithm was proposed.

2.2.2 Train Scheduling Using Simulation

Yalçınkaya and Bayhan (2012) created a framework for stochastic simulation modelling with the objective of finding a feasible timetable for all trains. Both arrival and departure times included in this study. In addition to that, with the real time data, this framework can be used for rescheduling as well. Which means proposed simulation model can deal with disturbance in the system. This paper gave very detailed information about model. The model created with ARENA discrete event simulation program. Assumptions of this model were similar to existing studies. A stochastic track failure model and blockage preventive algorithm were included as well. It has been shown that using simulation modeling for train scheduling is very effective because of the stochastic and complex nature of this problem.

Hassannayebi et al. (2019) suggested a simulation-based optimization approach for the timetabling problem. The modelling structure of this study is based on both response surface methodology and discrete event simulation. Several statistical tests were given. Simulation model was built by using MATLAB. Computational experiments were made using real-life high-speed railway data.

2.2.3 Train Scheduling Using Other Methods

Burdett and Kozan (2010) created a disjunctive graph model and a framework to solve train scheduling problem. An improved job shop approach was used with the objective of minimizing makespan. With the help of a few algorithms, problem was solved, and the results were presented in the study. Sotskov and Gholami (2012) also used disjunctive graph model to minimize total weighted tardiness of trains. In order to solve the model a shifting bottleneck algorithm was proposed and analyzed. Because of the complexity of the problem, a heuristic approach was presented as well.

Ying et al. (2020) modelled the problem as a markov decision process in order to minimize operating cost and passenger waiting time. The model was driven by stochastic customer demand. Deep reinforcement learning approach was used to solve the model. It was shown that this approach outperforms other classical metaheuristics.

Xie et al. (2020) proposed a mixed itinerary-size weibit model and an iterative method for timetabling and stochastic passenger assignment.

Abels et al. (2021) proposed a hybrid answer set programming (ASP) approach to solve a real-world scheduling problem which includes routing, conflict resolution and scheduling. Their aim was to minimize total delay time.

2.3 Papers on Rescheduling

Second type of papers is rescheduling/dispatching. These papers deal with the schedule of the trains after disruptions or disturbances occur. In a railways system it is very normal to have problems. In this thesis, these papers are categorized in three group: mathematical models, simulation models and other methods.

2.3.1 Train Rescheduling Using Mathematical Model

Meng et al. (2010) constructed a MILP model to tackle with this problem with the objective of making dwell time accord with planned time when there is a delay which means minimizing total delay time. An improved convergent particle swarm optimization algorithm was designed to solve the mathematical model. Results showed that proposed algorithm was quite capable of solving such problems. Acuna-Agost et al. (2011) also proposed a MILP model for this subject with the aim of minimizing delay costs. Some approaches such as local branching type cuts, right-shift rescheduling, and local search were suggested. In the same year, in another article, Acuna-Agost et al. (2011) proposed statistical analysis of propagation of incidents heuristic approach as well. Wang et al. (2012) created a fuzzy MIP model with the aim of both minimizing the number of trains affected and minimizing total delay. With the use of improved tolerance approach model was solved. Real-life experimental results were shown. Sato et al. (2013) proposed a MIP model to minimize passenger inconvenience. To address the model, cutting plane and shortest path algorithms were used in the paper. Li et al. (2014) suggested a MIP model with the aim of minimizing track changing cost and delay cost. A track-backup rescheduling approach was suggested. Stochastic nature of the problem was the one of main focus points the study. This algorithm assigns each train to a backup track based on the recovery time optimally. Also, a greedy algorithm was used in the process. Finally, extensive numerical experiments were conducted. Meng and Zhou (2014) proposed a MIP model. The big-M method and a lagrangian relaxation solution framework was used in order to solve the problem. A label correcting algorithm was added for each sub-problem. Also, a shortest path algorithm was used for each path finding sub-problems. In this model, rescheduling and rerouting made simultaneously. Kang et al. (2020) proposed a MILP model to minimize the traction idle time. They used a two-step sequential algorithm to solve the problem.

Zhan et al. (2016) formulated a MILP model with the objectives of minimizing train service deviations and minimizing train service cancellations for high-speed railways. A rolling horizon approach was proposed and tested on real-life instances. Fischetti and Monaci (2017) formulated a MILP model in order to solve a real-time rescheduling problem. An efficient heuristic was suggested and randomized variable fixing procedure was applied to improve results. Jiang and Zhou (2018) constructed two algorithms for this problem. First one is trying to minimize processing time and train operation time while second one is trying to minimize the difference between planned schedule and reschedule. A GA was used in order to solve the problem and, finally, model was tested. Wang et al. (2019) formulated a MIP model for this subject with the objective of minimizing total delays. Authors proposed a GA-based particle swarm algorithm to solve the mathematical model. Zhang et al. (2020) constructed a mixed integer nonlinear programming (MINLP) model with the objectives of minimizing maintenance cost and efficient operation. To be able to solve the model in a reasonable time, a Lagrangian relaxation-based decomposition algorithm was proposed. Also, the suggested non-linear model was converted into linear model by using linearization methods for efficiency of solution. Wang et al. (2021) examined a scenario where a metro line experiences a complete blockage and focuses on solving the problem of integrated train rescheduling and rolling stock circulation planning. A MILP approach and a two-stage heuristic approach was suggested. Their aim was to minimize the total weighted deviations.

2.3.2 Train Rescheduling Using Simulation

Almodovar and Rodenas (2013) proposed a discrete event-based simulation optimization approach in order to reassign trains with the aim of minimizing total time spent in system for passengers. The model was solved using two greedy heuristics. One is for on-line optimization and the other on is for improving the computational time. Near-optimal results were reached. Reassignment decisions were made by simulating the passenger flow. The proposed simulation model in this paper comprised of service model, demand model and discrete-event simulation model. Model was coded in MATLAB and tested on real life situations. It was concluded that for this model to be used in real time, computational time must be improved.

Shakibayifar and Sheikholeslami (2018) suggested a multi-objective simulation-based framework for rescheduling problem to minimize average delay time and minimizing total deviation. A neighborhood search heuristic was also used. This study presented a new stochastic simulation model for disturbance in the train schedule. Lastly, results were compared with the existing methods. After one year, Shakibayifar et al. (2019) proposed another multi-objective simulation optimization model with the same objectives.

2.3.3 Train Rescheduling Using Other Methods

Corman et al. (2010) proposed an alternative graph method to solve the rescheduling problem. Some strategies such as TS, rerouting and branch-and-bound were used. Model was tested on real life examples. Gholami and Krasemann (2018) modelled rescheduling problem as mixed graph and alternative graph. For re-routing and re-timing, a heuristic algorithm was proposed.

Zhu et al. (2020) suggested a reinforcement learning based rescheduling method. The proposed model learns how to make decisions about train rescheduling and can be applied online. Experiments were made and results were shown.

2.4 Analysis and Classification of the Literature Review

The literature review involved classifying the studies based on problem types (timetabling, dispatching, review), the structure of the studied roadway(single-track network, N-track), objectives, proposed models, and solution methods, as can be seen in Appendix A.1.

Research in the last decade has placed a strong emphasis on optimization techniques. Mathematical modeling and algorithmic approaches, particularly metaheuristic algorithms like GA, SA, and TS, have been widely applied to find efficient train schedules.

When reviewing the relevant studies, it is seen that the mathematical model approach has been extensively explored, whereas the hybrid approaches that combine metaheuristics with simulation methods has received less attention. This integration aims to capture the stochastic nature of railway systems while optimizing schedules effectively.

Additionally, among the limited number of papers including simulation, there is a tendency to focus more on the model construction aspect of the research. Furthermore, a gap in the literature is observed concerning potential simulation optimization approaches.

The significance of railway systems is growing day by day, and thanks to the advancements in computer technologies, the simulation optimization method has become a more crucial component in solving this problem, given its suitability for the problem's stochastic and complex nature.

Taking these factors into consideration, it is deemed appropriate in this thesis to work on the simulation optimization approach for the train scheduling problem.

CHAPTER THREE
A HYPOTETICAL TRAIN SCHEDULING PROBLEM

This thesis centers around the hypothetical train timetabling problem, TrainSchProb, proposed by Yalçınkaya (2010). In this section, a brief summary of this problem and the proposed solution is presented. More detailed information about the simulation model and the study can be accessed from PhD thesis of Yalçınkaya (2010) and Yalçınkaya, Ö., & Bayhan, G. M. (2012).

This chapter comprises two sections. The first section provides an overview of TrainSchProb, while the second section outlines the solution approach.

3.1 Railway Description

The problem's infrastructure follows a linear structure inspired by an actual railway line system. It includes an initial timetable plan where trains are exclusively scheduled to arrive and depart from the two terminal stations. The railway line, modeled after a real-world counterpart, resembles a single-track corridor, much like numerous lines found in both academic literature and actual railroad systems. Figure 3.1 displays visual representation of the single-track line and station infrastructure.

The corridor spans from east to west and encompasses a total of 10 real stations (ST). At the two ends of the single-line railway corridor, we have terminuses labeled as TS1 and TS10, denoting the starting and ending points, respectively, of the corridor. TS1 is located at the easternmost point, while TS10 marks the westernmost point. All stations along the corridor are furnished with 200-meter platforms to accommodate boarding and alighting events, while the overall length of the corridor spans 288,270 meters.

The train line is a single-track, and trains moving from the east and west can only pass each other at stations consisting of only two corridors. For a train to proceed to the next station, the corridor in between must be clear; otherwise, it must wait at the station.

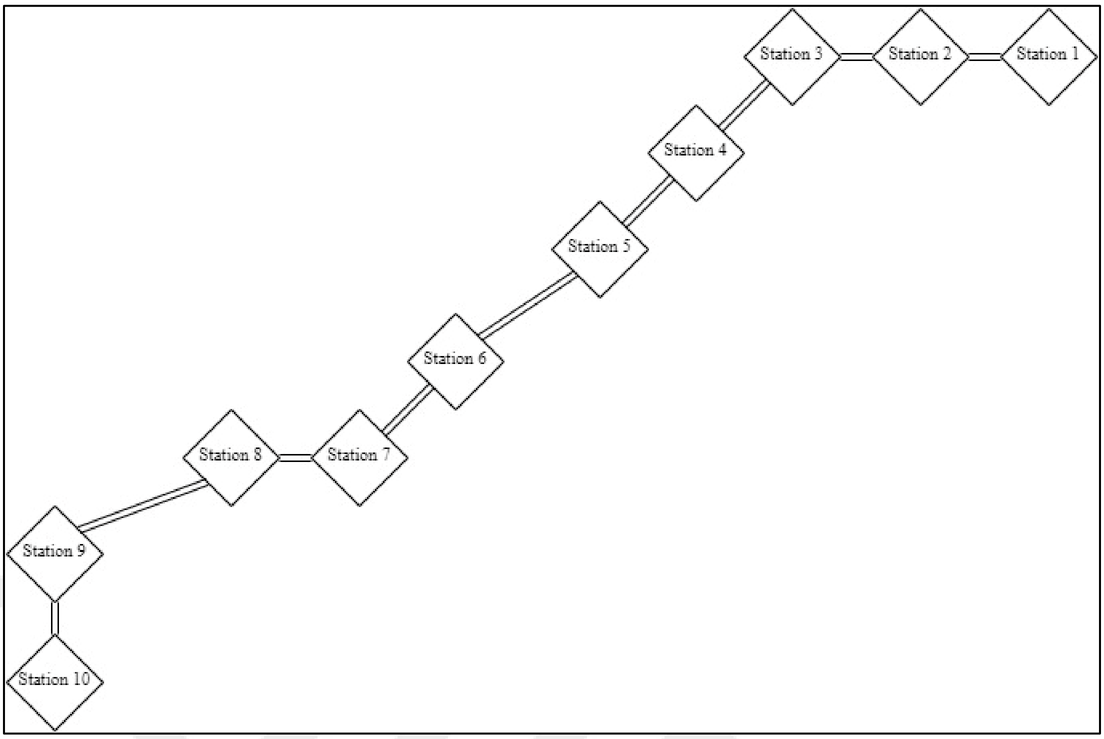

Figure 3.1 Diagram of railway corridor

3.2 Simulation Model for TrainSchProb

A simulation modeling framework is applied to this hypothetic problem with the aim of achieving a feasible schedule for the operational trains. In the following subsection this model will be summarized.

The planned train arrival and departure times for the initial stage are proposed by Yalçınkaya (2010) in Table 3.1. A total of 20 trains, 10 departing from the west (EB) and 10 departing from the east (WB), are assumed to operate in the system. The 10 trains departing from the west will travel eastward, while the 10 trains departing from the east will travel westward.

Additionally, it is assumed that all trains are of the same type and have a length of 50 meters. In this single-track railway system, the train track is bidirectional. It is assumed that there is a 200-meter platform at the stations for passenger boarding and alighting, with a maximum capacity of two trains at the same time.

Table 3.1 Planned train timetable

Station	Train	Planned Arrival Time	Planned Departure Time
Station1 (ST1)	WB_1	00:00	00:10
	WB_2	02:00	02:10
	WB_3	04:00	04:10
	WB_4	06:00	06:10
	WB_5	08:00	08:10
	WB_6	10:00	10:10
	WB_7	12:00	12:10
	WB_8	14:00	14:10
	WB_9	16:00	16:10
	WB_{10}	18:00	18:10
Station10 (ST10)	EB_1	00:00	00:10
	EB_2	02:00	02:10
	EB_3	04:00	04:10
	EB_4	06:00	06:10
	EB_5	08:00	08:10
	EB_6	10:00	10:10
	EB_7	12:00	12:10
	EB_8	14:00	14:10
	EB_9	16:00	16:10
	EB_{10}	18:00	18:10

In addition to these, some other assumptions have been added to make the model more effective. Some of these are as follows:

- To ensure the proper functioning of the model, dummy stations have been added between real stations, and the capacity of these dummy stations is a total of 1 train.
- The time it takes for trains to reach the parking area from terminal stations is not considered.
- Terminal stations have unlimited train capacity.
- The distance between terminal stations and parking areas is 100 meters.
- Parking areas for trains are unlimited.
- Deviation in the planned initial train timetable follows a uniform distribution with minimum and maximum values of -900 seconds and 900 seconds, respectively.

- Although there is a possibility of delays, the waiting time for trains at stations is 10 minutes and the delay time follows an exponential distribution with an average of 90 seconds.

- Trains are required to stop at each real station.

- If there is a track failure ahead of a train, it will make a stop at a dummy station.

- Trains traveling in opposite directions can only pass each other at the actual stations.

- Train speeds follow a uniform distribution with minimum and maximum values of 90 km/h and 110 km/h, respectively.

The stations serve as points where trains can halt for passenger boarding and alighting, parking, or waiting during repairs. Dummy stations (DS) are positioned along the tracks to accommodate train waits for track repairs. In Figure 3.2, a comprehensive illustration of the corridor from eastern park area to Station2 (ST2) is provided by Yalçınkaya (2010).

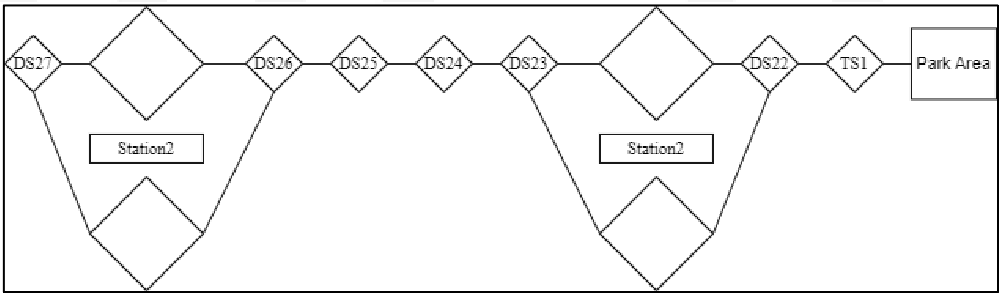

Figure 3.2 Detailed diagram of railway corridor

A simulation model for generating feasible schedules is constructed using ARENA Simulation Software 10.0. Initially, the railroad corridor is modeled, including its tracks, intersections and stations, with considerations for track failures and repairs. Subsequently, the logic for train flow within the corridor is simulated.

Since it is a single-track system with trains moving in opposite directions, more than one train may request to use the corridors between the stations at the same time. Choosing which train the corridor will be allocated to firstly emerges as a problem, and the way this allocation is done affects the performance of the system. Six different rules -first in first out (FIFO), last in first out (LIFO), shortest current traveling time (SCTT), longest current traveling time (LCTT), shortest remaining track part (SRTP), and longest remaining track part (LRTP)- are set to allocate tracks to candidate trains awaiting access at neighboring stations. The FIFO rule grants priority to the first train in line for the corridor. Similarly, the LIFO rule assigns priority to the last train in line for the corridor. The SCTT and LCTT rules allocate corridors based on the Current Travel Time attribute of the trains, which holds the value of their current travel times from the start. Finally, the SRTP and LRTP rules prioritize corridor allocation according to the Remaining Track Part attribute of trains, which contains the remaining track parts for each train.

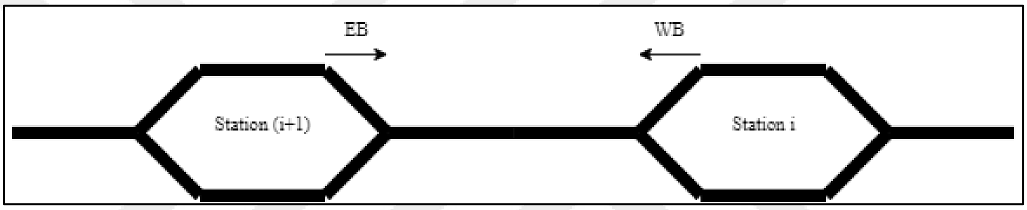

Figure 3.3 Illustration of possible corridor queues (1)

In Figure 3.3, a visual representation is provided to illustrate what has been described above. Moving trains in the system can queue up to use certain corridors along their routes. As seen in the figure, there can be situations where two trains moving in opposite directions.

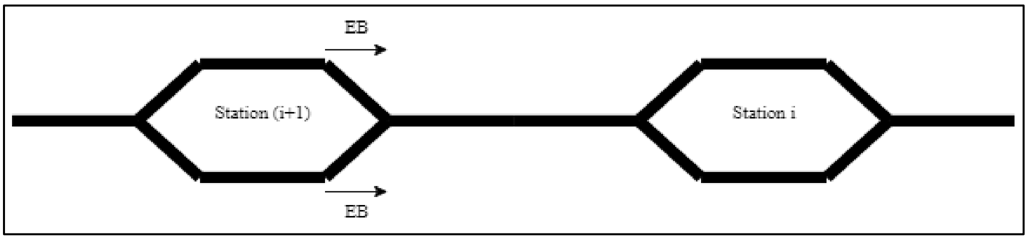

Figure 3.4 Illustration of possible corridor queues (2)

In figure 3.4, two trains moving in the same direction are in queue for the same track. This can occur due to some stochastic events in the system. Finally, multiple trains waiting to use the same corridor is illustrated in Figure 3.5 below. In all these cases, without creating a deadlock, priority is given to the trains waiting in the queue based on the rule applied for the corridor.

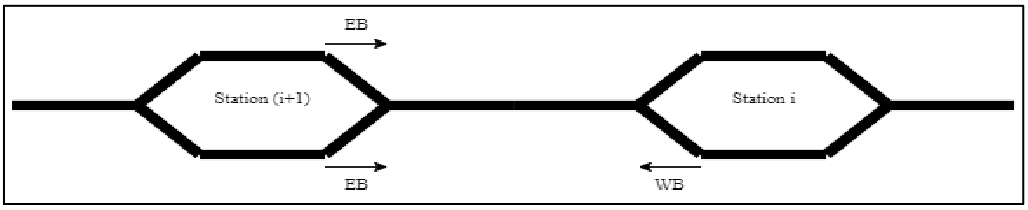

Figure 3.5 Illustration of possible corridor queues (3)

Also, there is a blockage preventive algorithm to avoid possible deadlocks. Every train movement is allowed after blockage preventive algorithm performs control on the three stations. By this way, all possible deadlocks are ruled out. In figure 3.6 shows an example of this procedure. As it can be seen, if WB trains in Station i and Station $(i + 1)$ move, it will create a deadlock in the system. So, these trains must wait until further notice.

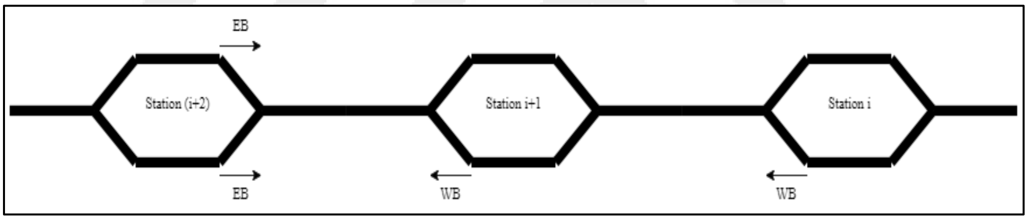

Figure 3.6 Illustration of deadlock prevention system

Train speeds are relaxed, and extra unforeseen delays at stations are introduced. Events related to track failure and subsequent track repair are implemented. A track failure is an occurrence that obstructs a train from utilizing a track for a journey. Once the track is repaired, the train can resume its trip. The probability distributions governing the durations of track failures and the subsequent repair times for the impaired tracks are presented in Table 3.2 below.

The first three columns provide information about the variable number, location, and length of the tracks. These tracks are sorted based on their lengths, with the shortest track ranked as 1. The ratios are calculated by dividing the lengths of these tracks by the length of this base track.

Table 3.2 Failure and repair time distributions

Track Part	Length (meter)	Rank	Ratio	Failure time distribution(sec)	Repair time distribution
ST1-ST2	27370	3	1.591	EXPO (54296)	EXPO (2864)
ST2-ST3	31900	6	1.855	EXPO (46586)	EXPO (3338)
ST3-ST4	28030	4	1.630	EXPO (53017)	EXPO (2933)
ST4-ST5	36610	7	2.128	EXPO (40592)	EXPO (3831)
ST5-ST6	44650	9	2.596	EXPO (33283)	EXPO (4673)
ST6-ST7	26800	2	1.558	EXPO (55451)	EXPO (2805)
ST7-ST8	17200	1	1.000	EXPO (86400)	EXPO (1800)
ST8-ST9	28900	5	1.680	EXPO (51421)	EXPO (3024)
ST9-ST10	42010	8	2.442	EXPO (35374)	EXPO (4396)

It is assumed that the failure time for the base track follows an exponential distribution with a mean of 86400 seconds, equivalent to 24 hours. Similarly, the failure times for the other tracks are assumed to follow exponential distributions with means of 86400 seconds divided by their respective ratios. Similarly, the repair time for the base track follows an exponential distribution with a mean of 1800 seconds and other parts are calculated by same logic.

In other words, the expected values for these tracks are inversely proportional to their lengths, meaning that longer track sections are more likely to experience track failures. In the simulation model, variables are used to manage track failures.

CHAPTER FOUR
AN OVERVIEW OF HEURISTIC ALGORITHMS FOR TRAIN SCHEDULING PROBLEM

The primary aim of the single-track train timetabling model presented in this thesis is to address train conflicts, ultimately achieving a predefined objective. However, finding the optimal schedule for trains operating on a single-track is a challenge that belongs to the NP-Hard complexity class, which means that when attempting to optimize the train schedule by determining the siding for each conflict, the computational complexity grows exponentially as the number of conflicts present in the schedule (Cai & Goh, 1994). This makes obtaining the best solutions for real-world challenges in a sensible timeframe challenging.

In practical scenarios for solving the train scheduling problem, finding the optimal solution may not be the primary objective since it requires a considerable amount of time and may not be feasible due to certain constraints. Therefore, there is a growing demand for heuristic techniques that can generate nearly optimal solutions within a short time frame. That is why, in this chapter, several metaheuristic algorithms commonly used in similar problems are presented.

4.1 Genetic Algorithm

The first publication on GAs was created by John Holland (1975), which marked the beginning of this field. GAs are an approach to problem-solving that takes inspiration from natural evolution and are designed to find satisfactory solutions to complex optimization problems that involve a large number of possible solutions. While it may not always find the best possible solution, it can identify good ones through a heuristic approach.

A GA operates by representing possible solutions to an optimization problem as a group of individuals forming a population. Each individual is defined by a chromosome, which is essentially a string of bits encoding a possible solution to the problem and elements of chromosomes are called genes. By utilizing genetic operators like selection, crossover, and mutation, the group of individuals evolves over time in pursuit of an optimal problem solution (Yu & Gen, 2010).

Haupt & Haupt (2004) outlined the following benefits of GAs:

- Ability to avoid local optimum,
- Ability to produce a set of optimal solutions, rather than just a single solution,
- Ability to handle a large number of variables,
- Requires only a single objective function to evaluate an individual's fitness,
- Can produce multiple optimal outcomes across various generations,
- Probabilistic and stochastic in nature.

In Table 4.1 pseudocode of the standard GA is exhibited.

Table 4.1 Pseudocode of GA

```
BEGIN
    INITIALIZE population with random possible candidate solutions
    EVALUATE fitness values of population
    REPEAT
        SELECT parents
        CROSSOVER
        MUTATE
        EVALUATE solutions
        REPLACE
    UNTIL termination criteria is satisfied
END
```

4.1.1 Elements of GA

GA is much more intricate than local search techniques, as it involves numerous interrelated components. This subsection will provide a more comprehensive explanation of these different components.

4.1.1.1 Representation

Representation is probably the most fundamental issue in GA. This stage establishes a connection between the initial problem context and the space where problem-solving will occur through process of evolution, paving the way for the algorithm. There can be many different encodings for the same problem. Each different encoding changes the shape of the search space and the approach to searching within it. Therefore, the type of encoding affects the required genetic operators for a successful optimization (Yu & Gen, 2010).

Encoding schemes vary depending on the specific problem being solved. There are several commonly used methods for encoding, including binary, octal, hexadecimal, permutation and value encoding.

In GAs, binary encoding is a frequently employed encoding scheme where each chromosome is depicted by a binary string consisting of 0s and 1s. This means that every chromosome is essentially a string of bits (Kumar, 2013). This approach offers faster execution for genetic operators. Nonetheless, it necessitates additional steps to convert the data into binary form, and the accuracy of the algorithm is reliant on the precision of this conversion (Katoch et al., 2021).

In Octal and Hexadecimal Encoding, chromosomes are represented using series of numbers ranging from octal numbers (0-7) and hexadecimal numbers (0-9, A-F), respectively (Kumar, 2013).

Permutation encoding is a commonly used method in problems that require ordering. It involves representing the gene or chromosome using a string of numbers that corresponds to the position of each element in a sequence. Permutation encoding is applicable only to problems that have a specific order. In order to maintain the consistency of the chromosome for such problems, certain types of crossover and mutation corrections must be made. Partially mapped crossover (PMX), order crossover (OX) and cycle crossover (OCX), are the three common crossover operators used on permutation encoding. Inversion is the most frequently used mutation operator on ordered chromosomes. It alters the gene's location (Kumar, 2013).

Value encoding is a method of representing each chromosome as a sequence of values. The values can be integers, real numbers, characters, or any other type of object relevant to the problem such as numbers, characters and complex objects.

4.1.1.2 Initial Population

One of the main issues to address is the population's size and the approach used to select individuals. The population size in GAs has been studied from various theoretical perspectives, however, the primary concept revolves around achieving an equilibrium between efficiency and efficacy. If the population is too small, there will be limited search space available, leading to the possibility of getting stuck at a local optimum. Conversely, if the population is too big, the search space is expanded, but it comes at the cost of increased computational complexity (Roeva, O et al., 2013). Hence, selecting a reasonable population size is crucial.

Reeves, C., & Rowe, J. E. (2002) mentioned the important points in the produced solutions are as follows.

- Chromosomes must be created to represent different points of the search space.
- Diversity must be preserved.
- One should be careful to avoid generating identical chromosomes.

4.1.1.3 Evaluation

Following the creation of an initial population, every chromosome undergoes an assessment to determine its fitness value. The evaluation process of GA can be performed using different methods, including a mathematical function or a simulation model.

4.1.1.4 Parent Selection

GAs include a crucial step called "parent selection" that decides whether a specific chromosome will be involved in the reproduction process or not. The speed at which a GA converges is significantly influenced by the selection procedure, which is determined by the selection method used. There are various widely used selection techniques such as roulette wheel, rank, stochastic universal sampling, elitism and tournament.

One of the most common ways to implement this is roulette wheel method where each individual is mapped onto a wheel proportional to its fitness score. The wheel is then spun randomly to choose which solutions will be selected for the next generation Jebari K. (2013). An illustration for this method is given in Figure 4.1.

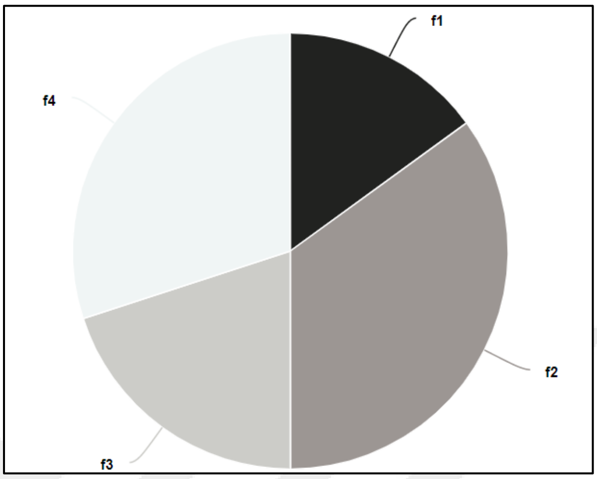

Figure 4.1 Illustration of roulette-wheel

Let's assume that the fitness values of P individuals are $f_1, f_2, f_3 \ldots f_P$. Then, the probability of choosing individual i can be formulated as follows:

$$\frac{f_i}{\sum_{k=1}^{P} f_k} \tag{4.1}$$

Rank selection is a variation of roulette wheel selection, in which ranks are assigned based on the fitness value of individuals instead of their actual fitness values. The individuals are given ranks, and then selected based on these ranks. This method helps to reduce the risk of getting stuck in a local minimum, thereby preventing premature convergence of the solution Jebari K. (2013). Also, his method allows for a better perspective on large differences in fitness values, and it eliminates the need for the exact fitness values of individuals, as only their ranking according to their quality is required.

K.D. Jong (1975) introduced elitism selection as a way to enhance the effectiveness of Roulette wheel selection. This method guarantees that the best individual of a generation will always be included in the next generation. If the fittest individual is not chosen during the normal selection process for the next generation, the elitist one will be automatically added.

Stochastic universal sampling (SUS) is a variation of the roulette wheel selection approach, with the added feature of using a random starting point in the list of individuals and selecting them at evenly spaced intervals. This ensures that every individual has an equal probability of being chosen for crossover in the next generation (Abdulal & Ramachandram, 2011).

Another option for selection that is not based solely on fitness is tournament selection, where a group of individuals is selected and assessed, and the fittest one is chosen to participate in reproduction. Tournament selection offers a potential advantage over other types of selection methods since it just requires a preference ranking among groups or pairs of strings. Thus, it can be applied in scenarios where lacking explicit objective function. The tournament selection method requires an additional parameter known as the tournament size, which typically has a value of two. This technique guarantees diversity as each individual has an equal opportunity of being chosen, even though this can potentially decrease the convergence rate (Hassanat et al., 2019).

4.1.1.5 Mutation

Mutation is a fundamental operation that creates unanticipated and unforeseeable changes in chromosomes to maintain the genetic diversity between generations. In the context of GAs, mutation plays a crucial role in reintroducing genes that were eliminated during the selection process, allowing them to be evaluated under a new context, as well as introducing new genes to prevent getting stuck in local optima (Gen et al., 2008).

There are several frequently employed mutation operators in GAs, including displacement, simple inversion, and scramble mutation. displacement mutation (DM) is a process that entails picking a random substring from an individual's solution and relocating it to another position, all while maintaining the validity of the resulting solution. Exchange and insertion mutations are variations of DM where a portion of the individual's solution is either swapped with another portion or inserted into a different location, respectively (Lim et al., 2017).

The simple inversion mutation (SIM) operator is responsible for reversing the substring between two specified locations in a chromosome. This inversion operator randomly selects a string and reverses it before placing it in a random location within the solution (Lim et al., 2017). On the other hand, the scramble mutation (SM) rearranges the genes within a specified range of a chromosome randomly. It then verifies if this new order improves the fitness value of the individual (Lim et al., 2017).

One of the most important GA parameters is mutation rate which specifies the number of chromosomes to be mutated in each generation. Mutation is used in GA to avoid getting stuck in local optima, but if it happens too frequently, it can turn GA into a random search instead of being an efficient evolutionary algorithm (Hassanat et al., 2019).

4.1.1.6 Crossover

Crossover is one of the primary genetic operators that operates on two individual solutions concurrently and produces offspring by merging the genes of both chromosomes. Different crossover techniques are developed to obtain the best possible solution in the minimum number of generations. One of the simplest methods is one-point crossover, which involves dividing the parental chromosomes at a single point and combining them to produce an offspring (Umbarkar & Sheth, 2015). Another technique is k-point crossover, where two offspring are created by merging the parents at k crossover points (Umbarkar & Sheth, 2015).

In uniform crossover is another widely used method, which treats each gene separately without dividing the chromosome into segments like other techniques. In uniform crossover, a coin is flipped for each gene to determine its inclusion in the offspring (Umbarkar & Sheth, 2015).

The masked crossover (MX) operator employs a mask vector to identify the bits of the parents that are passed on to the offspring. Initially, the bits of the parents are duplicated. The bits from the first parent are transferred to the first offspring, while the bits from the second parent are assigned to the second offspring. Next, the offspring swap bits with each other only at the positions where the mask vectors of the parents were 1.

A crossover technique known as partially matched crossover (PMX) was introduced by Goldberg and Lingle in 1985. This process involves transferring both the order and value information from the parent to their offspring. The genetic material from one parent is contributed to the offspring, while the corresponding material from the other parent is scattered throughout the child. Afterward, the remaining genetic material is directly copied from the second parent.

The crossover rate refers to the probability of offspring being generated per generation relative to the population size. This rate determines expected number of chromosomes that will experience the crossover operation. An increased crossover rate facilitates the exploration of a wider solution space. Additionally, it lowers the likelihood of converging to an incorrect optimal solution. However, if the rate is excessively high, it can lead to significant wastage of processing time (Gen et al., 2008).

4.1.1.7 Replacement

To generate a new generation, it is essential to implement a replacement strategy that determines which individuals remain in the population and which are substituted by offspring. The new generation can consist of chromosomes from the previous generation, as well as chromosomes generated through mutation and/or crossover operations. A commonly employed replacement strategy is elitism, which ensures that a certain number of the most exceptional individuals survive in each generation, so the last group of chromosomes includes the optimal solution discovered to date (Coley, 2003).

4.1.1.8 Termination Criteria

Other search methods may halt when they identify a local optimal solution, but GAs are probabilistic searches that have the potential to continue indefinitely. In practice, a stopping criterion is necessary, such as constraining the number of fitness calculation or computer time (Haupt & Haupt, 2004).

There are three conventional termination criteria (Michalewicz & Michalewicz, 1996):

- A maximum number of generations.
- The maximum number of fitness function calculations.
- The low probability of making substantial improvements in the upcoming generations.

4.1.2 GA Parameters

As mentioned in the previous sections, GA consists of four main parameters which are Crossover rate, mutation rate, Population size and iteration number. Identifying how different parameters of the GA interact with each other can significantly affect the quality of the solution and maintaining a balance between parameter values can lead to an improved solution for the GA (Hassanat et al., 2019).

Changing the values of these parameters, either by increasing or decreasing, can have a positive or negative impact on the outcome of the GA, and selecting appropriate parameters is a challenging task that requires careful consideration. This is because the parameters are interconnected and altering one can have a ripple effect on the others. Research has shown that finding the right balance of parameters is crucial for obtaining optimal results (Davis, 1991).

4.2 Tabu Search

In 1986, a new method called "tabu search" was proposed by Fred Glover with the aim of helping hill climbing algorithms overcome local optima. Glover (1989) described TS as an adaptive algorithm that extends iterative methods such as SA. TS has the ability to incorporate diverse techniques such as linear programming algorithms and specialized heuristics that are directed towards circumventing the limitations imposed by local optima.

Tabu search operates by continuing with the search, even when a local optimum is encountered, through the inclusion of non-improving moves. It uses a memory mechanism known as tabu list to avoid reexploring the same results. Pseudocode of TS can be seen from Table 4.2.

Table 4.2 Pseudocode of TS

```
GENERATE initial solution
DEFINE tabu list size, termination criterion and aspiration criterion
SET best solution, current solution, tabu list
REPEAT
    FOR neighborhood size
        GENERATE new solution
        EVALUATE new solutions
        CHOOSE best solution among all candidates
        MOVE current solution
        IF current solution is better
            UPDATE best solution
        END
        UPDATE tabu list and aspiration criteria
    NEXT
UNTIL stopping criterion is met
```

The central idea of TS is selecting the neighboring solution with the lowest cost increase. It employs a memory-based mechanism to store previously explored solutions, preventing revisits to them. Memory-based techniques are an essential feature of TS algorithms. Although some applications of TS may necessitate advanced functionalities, basic designs can often produce satisfactory results.

TS is applied in various fields, such as scheduling, computer channel balancing, cluster analysis and assignment. Also, it is utilized to solve numerous technical problems, including graph coloring, character recognition, and the traveling salesman problem (Jaziri, 2008).

4.2.1 Elements of TS

The components of TS can be summarized as follows.

4.2.1.1 Neighborhood Generation

To optimize the function across the solution space, it is necessary to define a framework in the proximity of the space of potential solutions and the starting point. The exploration process then proceeds by modifying the current solution. The best move is chosen from neighborhood. The algorithm then moves to the subsequent iteration, looking for a solution within the neighborhood of the accepted move.

4.2.1.2 Tabu List

In TS, the tabu list stores information about the previously explored solutions. It is continuously updated as the search progresses and contains the most recent moves. The tabu list assists in directing the search from the present solution to the following one. During every iteration, the tabu list gets refreshed to support the ongoing exploration procedure, and it prevents revisiting recently explored neighbors, which can save computational resources.

The appropriate size of the tabu list for good results usually increases with the problem’s growing size, but there is no single rule for determining the best size for every type of problems. To determine an effective tabu list size for a specific problem, it is advisable to monitor cycling. The best sizes typically fall within an intermediate range, and it is recommended to allow the size to vary within this range.

4.2.1.3 Aspiration Criteria

Tabus can sometimes be too restrictive. They can prevent beneficial moves from being made, even when there is no risk of looping, or they can result in the search process stagnating overall. To address this, algorithmic tools called aspiration criteria must be employed to revoke or cancel tabus.

4.2.1.4 Termination Criteria

Theoretically, the search process could continue indefinitely, unless the best solution for the problem being addressed is known. However, it's necessary to halt the search within a reasonable timeframe. Commonly used stopping conditions in TS include the following.

- Predetermined iteration limit
- Predetermined number of iterations without an improvement in the objective
- Predetermined objective function value

4.3 Simulated Annealing

SA was first introduced in 1982 by Kirkpatrick et al. SA is a widely used approach for solving combinatorial optimization problems, particularly those that involve minimizing functions with a large number of variables. SA involves applying a sequence of gradual temperature reduction procedures to address nonphysical optimization problems, which have the capability of converting an unordered and suboptimal solution to a highly optimized and desirable one. This method provides an interesting physical analogy for problem optimization, and also has the potential to transfer mathematical insights from physics to practical optimization issues.

SA uses a strategy that is comparable to iterative improvement, with the primary distinction being that annealing allows for controlled upward perturbations. The objective is to obtain a good solution. By allowing moves to transform a configuration into a worse one, SA can escape local minima and discover a better path. Nevertheless, due to the careful regulation of uphill movements, there is no danger of unintentionally transitioning to a significantly worse solution after approaching a favorable final solution (Rutenbar, 1989).

SA starts with a randomly generated solution. Then, a new solution is produced by perturbating current solution within the neighborhood. If the newly generated solution is an improvement, it is accepted. However, if it is worse, SA employs a stochastic acceptance criterion to decide whether to accept it or not. The temperature parameter in SA is gradually decreased over time, which reduces the likelihood of accepting solutions with poorer objective function values. As a result, after some time, SA transitions into a local search algorithm. Finally, after a predetermined number of iterations, the algorithm terminates.

Pseudocode for SA can be seen in Table 4.3.

Table 4.3 Pseudocode of SA

GENERATE initial solution
INITIALIZE R_{max} (predetermined number of iterations) and E (temperature parameter), E_0 = stopping temperature
WHILE E > E_0 **DO**
FOR iteration number < R_{max}
Compute new solution
Compute Δ = current solution - best solution
IF $(\Delta < 0)$ or stochastic acceptance is met **THEN** update current solution
NEXT iteration number
Reduce E
END WHILE

4.3.1 Elements of SA

Four basic components of SA were mentioned by Rutenbar, R. A. (1989), as follows.

4.3.1.1 Configuration

Represents the potential problem solutions that will be explored in search of a satisfactory answer. It is crucial for obtaining meaningful results. The method for representing solutions is specific to the problem when dealing with combinatorial or permutation optimization problems.

4.3.1.2 Move Set

The set of feasible moves should be easy to calculate and allow us to access all possible configurations. These moves are the computations we perform to transition between configurations during the annealing process.

Simulated annealing's effectiveness is greatly impacted by the selection of the neighborhood function and generation of new solutions should incorporate minor random variations and enable the exploration of all potential solutions (Moscato, 1994).

4.3.1.3 Cost Function

To evaluate the quality of a particular configuration. It is usually problem specific. In this thesis, simulation results will be used for cost function.

4.3.1.4 Cooling Schedule

To use the annealing process to transform a random solution into a desirable one, we need a starting temperature as well as guidelines for decreasing the temperature, determining the amount of temperature reduction, and deciding when to stop the annealing process.

In general, the initial temperature (E) is determined to ensure that the proportion of unfavorable moves accepted is equivalent to a specific value. Ben-Ameur, W (2004) showed that the acceptance probability function is convex for low temperatures and concave for high temperatures.

Two categories of cooling schedules exist, namely, static schedules and adaptive schedules. Strenski and Kirkpatrick (2000) showed that he most effective cooling schedules are not necessarily strictly decreasing in temperature. Additionally, their findings demonstrate that, for the test problem involving a surface with white noise, geometric and linear cooling schedules yield better results than inverse logarithmic cooling schedules, provided that there is enough computation power.

Kirkpatrick, Gelatt, and Vecchi (1983) proposed a cooling schedule that is commonly used in the literature, which includes three parameters: initial temperature E, temperature decrease function and minimum temperature E_0.

To ensure that any newly generated solution during a state transition is accepted with a high probability close to 1, the initial temperature E value should be set to a sufficiently high value.

Several techniques and algorithms have been utilized to ascertain the parameters of the simulated annealing cooling schedule in various optimization scenarios. Equation 4.2, also known as geometric cooling schedule, was proposed by van Laarhoven and Aarts (1987).

$$E_t = \alpha^t * E \tag{4.2}$$

The temperature values of the temperature at iteration t and initial temperature are represented by E_t and, E respectively. The cooling factor, α, is a constant value chosen from the range of 0.8 and 0.99.

The linear cooling schedule is defined by Kirkpatrick et al. (1983) and is given by Equation (3):

$$E_t = E - \alpha * t \tag{4.3}$$

Here, "t" represents the iteration number and "α" is the decay parameter that determines the rate of decrease.

Another commonly used cooling schedule in literature is logarithmic cooling schedule. It is defined by Geman & Geman (1984) as given by Equation (4):

$$E_t = \frac{k}{\log(1 + t)} \tag{4.4}$$

The cooling schedule described by the logarithmic function is defined by initial temperature E, a positive constant "k" and number of iterations "t". This approach takes a long time to converge and demands significant computation resources.

These approaches may not ensure convergence to the global optimum, but they converge faster towards a high-quality solution.

CHAPTER FIVE

SIMULATION OPTIMIZATION FOR TRAIN SCHEDULING PROBLEM

Yalçınkaya (2010) proposed four simulation integrated GAs for the TrainSchPrb problem, three of which were hybrid algorithms created with local search algorithms. These three hybrid algorithms were as follows:

- Local Search on the population's best individual (SimGAb)
- Local Search on the population's first two best individuals (SimGAfs)
- Local Search on the population's best and worst individuals (SimGAbw)

These three hybridization methods were implemented to proposed genetic algorithm (SimGA). Finally, the results of the proposed algorithms are presented. In this thesis, various methods and experiments are used to achieve better results for the TrainSchPrb.

This chapter introduces four simulation-based algorithms aimed at producing a feasible train timetable while optimizing the average travel time for trains. The chapter is divided into six subsections. The first subsection presents an analysis of previous results. In the second subsection, a new simulation optimization algorithm (SimNGA) is proposed. The third subsection introduces a simulated annealing algorithm (SimSA) for the problem. In the fourth subsection, a TS algorithm (SimTS) is presented. The fifth subsection details a hybrid approach that combines GA and TS (SimHA). Lastly, the fourth subsection provides the outcomes of the algorithms and discusses the findings in detail.

It is crucial to mention the analysis conducted on the results of Yalçınkaya (2010) and the conclusions derived from it before introducing the proposed methods. These conclusions have led to variations in the simulation integration and evaluation of the results, which will be elaborated in subsection 5.1.

5.1 Examination of the TrainSchProb

At the beginning, an examination of the results revealed that despite testing different algorithms and parameters, the outcomes quickly converged to a similar value, raising suspicions that the algorithm might be trapped in a local optimum. However, it was also noticed that different solutions could yield the same result, as evidenced by various examples in Table 5.1. Therefore, it was considered that the variables being optimized may or may not have statistical significance due to the given system parameters.

After considering these reasons, it was understood that to achieve more accurate results, further investigation was necessary into the simulation environment, problem parameter values, and the stochastic nature of the problem.

The hypothetical problem under study involves a high level of stochasticity to enhance realism. Various chance-based events, such as track failure times, track repair times, train departure times from terminals, and train waiting times at stations, contribute to this stochastic nature. However, since these events do not occur frequently, obtaining a realistic result with a low number of replications can be challenging. To address this, the number of replications was increased to determine if the result obtained with 20 replications was statistically accurate.

In the conducted studies, it was observed that the solution found increased as the replication number increased. This suggests that the true value is likely greater than the optimum value of 20742.93 found during the optimization. One possible reason for this discrepancy could be the inclusion of rare probabilities that may not be adequately evaluated with a small number of replications.

The examinations conducted revealed that when evaluating with 20 replications, two distinct solutions that produced the same outcome had varying values as the replication number was increased. An illustration of this is provided in Table 5.1.

As seen in Table 5.1, chromosomes with very different configurations have the same fitness value. However, it is observed that the result changes when the replication number is increased.

Table 5.1 Results of some experiments

	Method	20 replication fitness value	100 replication fitness value	200 replication fitness value
Chromosome 1 (6,3,4,2,2,4,4,1,4)	SimGA	20742.93	21343	21427
Chromosome 2 (6,3,1,2,5,1,4,6,4)	SimGA	20742.93	21328	21481
Chromosome 3 (6,3,5,2,2,1,2,6,4)	SimGAb	20742.93	21317	21409
Chromosome 4 (1,6,4,2,2,5,2,3,5)	SimGAb	20742.93	21239	21360
Chromosome 5 (6,3,5,2,2,1,4,1,4)	SimGAfs	20742.93	21343	21427
Chromosome 6 (6,6,5,2,2,1,5,6,4)	SimGAfs	20742.93	21316	21414
Chromosome 7 (1,3,1,2,4,4,4,1,4)	SimGAbw	20742.93	21273	21422
Chromosome 8 (1,3,1,2,4,1,4,6,2)	SimGAbw	20742.93	21335	21443

Furthermore, we can test whether these values are statistically different from each other by conducting a paired t-test which is a statistical technique that is used to determine whether there is a significant difference between the means of two related groups of data. To demonstrate let's consider the Chromosomes 1 and Chromosome 2. With %95 confidence interval (significance level 0.5), Null (H_0) and Alternative (H_0) hypotheses can be expressed as:

$$H_0: population\ means\ of\ Chromosome 1\ and\ Chromosome\ 2\ are\ equal$$

$$H_1: population\ means\ of\ Chromosome 1\ and\ Chromosome\ 2\ are\ not\ equal$$

The results from Output Analyzer tool can be observed from Figure 5.1 and Table 5.2. For this test, 500 replication is used.

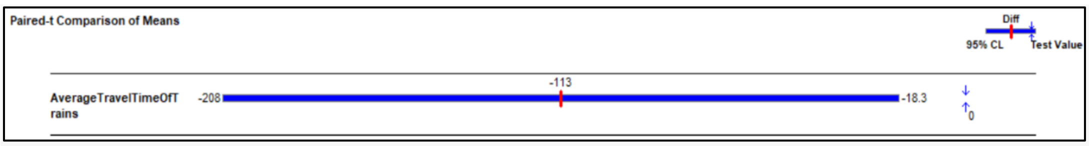

Figure 5.1 Mean difference between Chromosome1 and Chromosome2

Table 5.2 Results of output analysis

Paired- Means Comparison: Chromosome1-Chromosome2

IDENTIFIER	Estd. Mean Difference	Standard Deviation	0.950 C.I. Half-Width	Minimum Value	Maximum Value	Number of Observations
AverageTravel	-113	1.08e+003	94.8	1.8e+004	3.873+004	500
TimeofTrains				1.8e+004	3.873+004	500
	Reject H_0 => Means are not equal at 0.05 level					

By rejecting the null hypothesis H_0, it was concluded that the mean of these two values is statistically different from each other. As it can be seen from the test conducted, although the chromosomes seem to have the same fitness value in 20 replications, they actually produce different results.

After obtaining these results, a study was conducted to determine the adequate number of replications to be used in the algorithm. By considering the half-width value and conducting several tests, it was found that 500 replications were statistically sufficient. However, setting the simulation replication number to 500 for the simulation integrated algorithm to be implemented ceases to be an option due to the unacceptable increase in the computational time of the algorithm. That is why, algorithms will be run with 20 replications, but the fitness values of the best chromosomes obtained will be obtained by running the simulation program with 500 replications for these configurations. The procedures to be implemented can be summarized as follows:

i. The proposed algorithm will be implemented using the built-in Visual Basic tool in ARENA Software.

ii. Simulation integrated algorithm will be run with 20 replications.

iii. Results will be collected.

iv. To obtain more accurate fitness values, the best chromosome configuration will be simulated for 500 replications.

In the light of these results and data, evaluations are made according to this procedure. In this way, more accurate results will be obtained and an approach that is most suitable for the problem will be determined.

5.2 SimNGA for TrainSchProb

In this thesis, a new genetic algorithm SimNGA is proposed for TrainSchProb. In this subsection details of proposed SimNGA are presented.

In this thesis, the proposed SimNGA algorithm differs from the SimGA presented in the mentioned dissertation in some respects. In this subsection, the resemblances, and distinctions between these two algorithms will be discussed. The reason for proposing SimNGA in this research is to improve the performance of the algorithm and reduce the risk of getting stuck in local optima. The brief comparison between the two algorithms is summarized in Table 5.3 below.

Table 5.3 Comparison of SimGA and SimNGA

	SimGA	SimNGA
Objective	Single Objective	Single Objective
Encoding	Permutation Encoding	Permutation Encoding
Selection Operator	Rank Selection	Tournament Selection
Crossover Operator	Single Point Crossover	Uniform Crossover
Mutation Operator	Displacement Mutation	Displacement Mutation
Replacement Strategy	Elitism Selection	Elitism Selection

5.2.1 Encoding and Decoding

In the proposed algorithm, each chromosome (solution) consists of nine genes that correspond to nine decision points and indicate dispatching rules. In the TrainSchProb, between the real stations, there are a total of nine main track segments. Consequently, each chromosome in the proposed GA consists of nine genes. For instance, 9^{th} gene in the chromosome corresponds to the dispatching rule utilized for candidate trains queued on the track between Station9 (ST9) and Station10 (ST10) and so forth. Visual representation of a chromosome can be seen in Table 5.4 below.

Table 5.4 Visual representation of a chromosome

Decision point	1st	2nd	3rd	4th	5th	6th	7th	8th	9th
Track	ST1-ST2	ST2-ST3	ST3-ST4	ST4-ST5	ST5-ST6	ST6-ST7	ST7-ST8	ST8-ST9	ST9 - ST10
Chromosome	Gene 1	Gene 2	Gene 3	Gene 4	Gene 5	Gene 6	Gene 7	Gene 8	Gene 9

Each gene in the chromosome is assigned a value within the range of (1, 6), representing a specific dispatching rule. For instance, the value 1 stands for the first in first out (FIFO) rule, while the value 2 indicates the last in first out (LIFO) rule. The values 3 and 4 represent the shortest and longest current travelling time (SCTT and LCTT) rules, respectively. The values 5 and 6 correspond to the shortest and longest remaining track part (SRTP and LRTP) rules.

5.2.2 Initial Population

To promote diversity, the chromosomes (Ch) in the initial population are generated randomly. Once an individual is generated, the dispatching rules in the model (R) are rearranged based on the gene (g) values within the chromosome. Subsequently, the simulation model is executed, and the average train travel time is calculated and recorded as the fitness value. The process for generating the initial population and evaluating it is detailed in Table 5.5 below.

Table 5.5 Procedure for initial population

```
Begin SimNGA
Generate Initial Population
        popSize: Population Size
        For p = 1 To popSize
                Generate a random solution (chromosome)
                Ch (p): p^th chromosome in population
                        For i = 1 To 9
                                g_i: the gene associated with the decision point = {1,2,3,4,5,6}
                        Next i
                Ch (p) = [g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, g_9] (A random solution is created)
                Reconstruct the simulation model
                        For i = 1 To 9
                                If g_i = 1 set R (i) meaning FIFO
                                If g_i = 2 set R (i) meaning LIFO
                                If g_i = 3 set R (i) meaning SCTT
                                If g_i = 4 set R (i) meaning LCTT
                                If g_i = 5 set R (i) meaning SRTP
                                If g_i = 6 set R (i) meaning LRTP
                        Next i
                Run the reconstructed model for 20 replications
                Measure the fitness value
        Next p
Rank the initial population
Record the best chromosome
```

5.2.3 Genetic Operators

The use of effective genetic operators is crucial in successfully addressing the problem and producing high-quality individuals in the population while preserving adequate diversity. These operators can be broadly classified into three categories: parent selection, crossover and mutation.

5.2.3.1 Parent Selection

The selection operator in GA is utilized to pick individuals based on their fitness level. In this thesis, tournament selection operator is adopted to avoid getting stuck in a local optimum. Tournament selection is a process in which several individuals (based on the tournament size) are chosen randomly from the population and then the one with the highest fitness is selected. An attempt is made to achieve a balance between exploration and exploitation by utilizing this method. This procedure can be seen in Table 5.6 below.

Table 5.6 Procedure for parent selection and crossover

genN: Generation number
maxGen: Maximum generation number
popSize: Population size
For genN = 1 To maxGen
    CROSSOVER
    crR: Crossover rate
    nCr = popSize * crR
        Select the parents with tournament selection
        For i =1 To nCr-1
            Parent(i): i^{th} chromosome in the population
            r1, **r2**: Random numbers between 1 and popSize
            If fitness (r1) < fitness (r2)
                Parent(r1) chromosome selected
            Else
                Parent(r2) chromosome selected
            End
        Next i
        Apply the uniform crossover to obtain children
        Templ Array: randomly generated binary array
        Child(i): i^{th} child from Parent(i) and Parent(i+1)
        For i = 1 To nCr-1
            Generate Templ Array randomly
            Assign gene values to Child(i) and Child(i+1) accordingly
        Next i

5.2.3.2 *Crossover*

Numerous crossover operators are available in GAs, which are utilized in various applications. The choice of crossover operator depends primarily on the encoding type used in the GA. It is crucial to consider both global convergence and search space while choosing crossover operators (Umbarkar & Sheth, 2015).

This thesis uses the uniform crossover operator to address the issue of diversity. Two offspring are produced from the chosen parents and the process continues based on a predefined crossover rate. A random selection process is utilized to determine whether each gene will be included in the offspring or not. The procedure for crossover can be seen from Table 5.6 and illustration of uniform crossover is given in Table 5.7.

Table 5.7 Illustration of uniform crossover operator

	Gene 1	**Gene 2**	**Gene 3**	**Gene 4**	**Gene 5**	**Gene 6**	**Gene 7**	**Gene 8**	**Gene 9**
Parent 1	3	5	3	1	2	4	5	6	2
Parent 2	1	4	2	3	5	3	4	1	6
Templ. Array	**1**	**1**	**0**	**1**	**0**	**1**	**0**	**0**	**1**
Offspring 1	3	5	2	1	5	4	4	1	2
Offspring 2	1	4	3	3	2	3	5	6	6

5.2.3.3 *Mutation*

The mutation operator is detailed in Table 5.8 below.

The first step involves selecting a child at random for process. Then, a random gene (g) in the child is selected, and its value is substituted with another potential value from the set N = {1,2,3,4,5,6}, while making sure the new value is not same with the current one. After this, the simulation model is rearranged, and the fitness of the child is evaluated by running the model for and computing the average train travel time value. The mutation process continues based on the predetermined mutation rate.

Table 5.8 Mutation procedure

MUTATION
mtR: mutation rate
nMt: the number of chromosomes selected for mutation
nMt = mtR * popSize
For i = 1 To nMt
Select a child randomly
Child(i): selected child
Child(i): [$g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, g_9$]
Randomly select a gene to mutate
$\boldsymbol{g_j}$: the selected gene
If $g_j = 1$, change g_j with a random value = {2,3,4,5,6}
If $g_j = 2$, change g_j with a random value = {1,3,4,5,6}
If $g_j = 3$, change g_j with a random value = {1,2,4,5,6}
If $g_j = 4$, change g_j with a random value = {1,2,3,5,6}
If $g_j = 5$, change g_j with a random value = {1,2,3,4,6}
If $g_j = 6$, change g_j with a random value = {1,2,3,4,5}
Reconstruct decision point in the simulation model
Calculate the value of fitness
Next i
Rank the population

5.2.3.4 Termination Criteria and Replacement Strategy

The termination and replacement strategy are explained in Table 5.9. The algorithm can be halted using the termination criteria approach if the best fitness value remains the same for a consecutive number of previous generations as predetermined.

The replacement strategy involves removing worst individuals from the population and adding new children. Then, to create the next generation, the parents and children are ranked based on their fitness values.

Table 5.9 Procedure for termination criteria and replacement strategy

Check termination Criteria
TermCt: termination counter
maxTermCt: maximum termination counter
bFtv(GN): the best fitness value of the generation
If bFtv(GN) = bFtv(GN-1)
TermCt = TermCt +1
If TermCt < (maxTerCt -1) Go To **Next Generation**
Else go to Stop SimNGA
Else TermCt = 0
Arrange next generation
Next Generation
Stop SimNGA

Lastly, the complete SimNGA framework for the hypothetical TrainSchPrb is presented in Table 5.10 below.

Table 5.10 Framework of SimNGA

Start SimNGA
Initialize population
Rank the population
Record the best fitness
genN: Generation number
maxGen: Maximum number of generations
    For genN = 1 To maxGen
        Perform Crossover
        Perform Mutation
        Rank the population
        Record the best fitness
        Check Termination Criteria
    Next genN
Stop SimNGA

5.2.4 Parameter Tuning and Computational Results for SimNGA

The quality of the solution in GA is directly affected by the interactions among its different parameters. Maintaining a balance among parameter values can improve the solution of the GA. There are four fundamental parameters in GA that are crucial for its performance which are population size, crossover rate, mutation rate and maximum generation number.

Calibrating parameters and conducting computational experiments are crucial in developing any algorithm. The initial step in computational experimentation is to identify a suitable set of benchmark instances. These instances can be obtained from other researchers or created and should include a range of sizes and difficulties with some indication of their complexity. These collections should then be divided into two subsets, with one subset designated for the algorithm and calibrating its parameters, while the second subset reserved for the conclusive computational tests. A larger set of instances is required to calibrate methods that have multiple parameters, as compared to those that have few parameters, to ensure their robustness. The calibration process must involve multiple phases, such as (Crainic et al., 1993):

I. Conduct exploratory tests to determine appropriate parameter ranges for the method by running the heuristic with various parameter configurations.

II. Determine the values of the parameters that exhibit robustness, meaning that they do not appear to significantly affect the algorithm's performance.

III. Carry out a systematic test to evaluate the other parameters. Generally, this is done by testing one parameter at a time while keeping the other parameters fixed at reasonable values.

The population size (PS) in GA represents overall count of individuals in the population. Choosing an appropriate population size is crucial, as a small population size would limit the search space and lead to convergence to a local optimum. Conversely, a very big population size would extend the search space but also increase the computational load, which can be impractical (Gotshall & Rylander, 2002). Therefore, it is essential to choose a reasonable population size that balances the trade-off between search space and computational efficiency.

The evaluation of how population size affects the algorithm's performance, as shown in Figure 5.2, reveals that the simulation integrated model results in a runtime that is unacceptably long for values exceeding 30. Thus, considering the reasons mentioned above, previous studies in the literature and the nature of the problem that we are dealing with, in this thesis, the values of 10, 20, 30, 40 and 50 for the PS are tested.

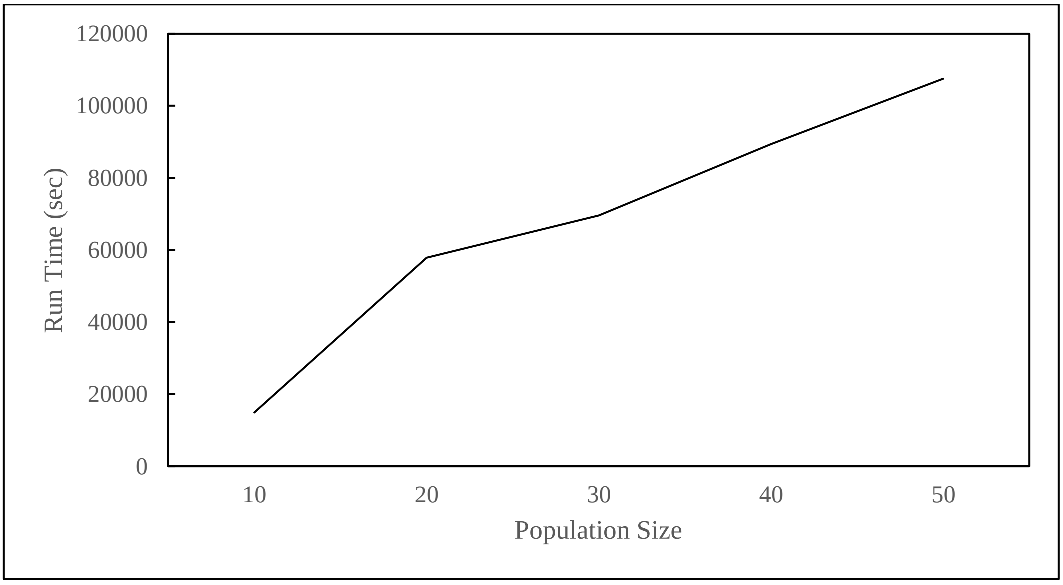

Figure 5.2 The effect of PS on performance for SimNGA

Furthermore, as shown in Figure 5.3, it is observed that the optimal result was achieved at each population size, but the population converged to the optimal value most rapidly when the population size was set to 30. Although values of 40 and 50 yielded good results, they were not included in the systematic tests due to significantly extending the runtime.

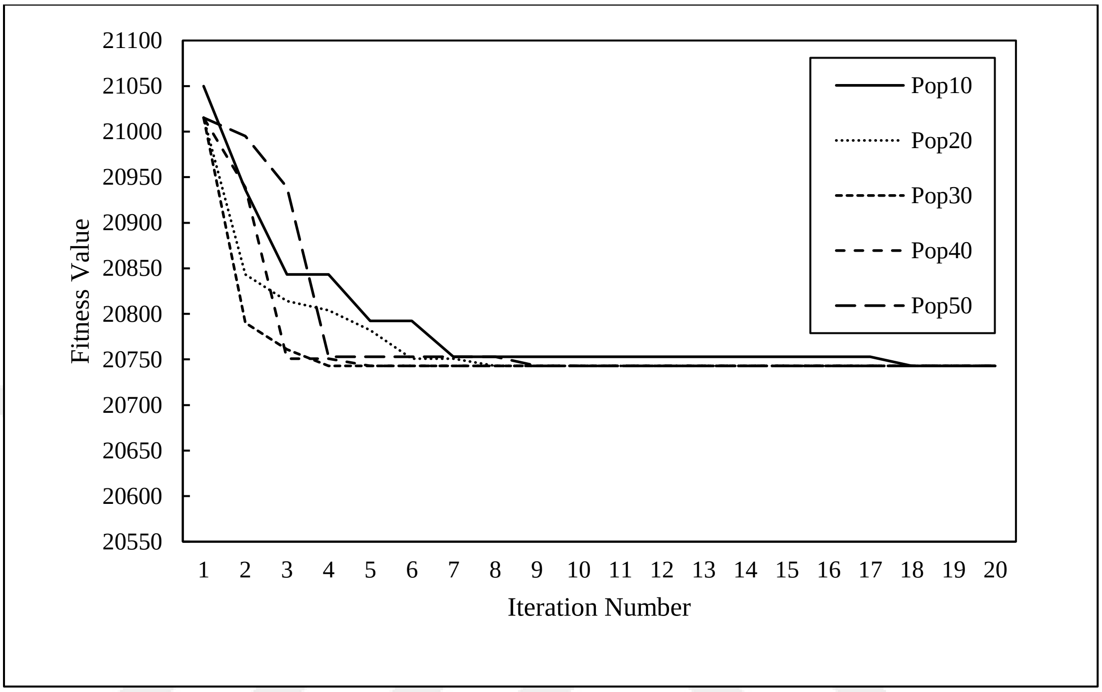

Figure 5.3 The effect of PS on convergence for SimNGA

The crossover rate (CR) refers to how often a crossover happens for chromosomes within one generation. A crossover rate of 100% would mean that all offspring are produced through crossover, while a 0% crossover rate would mean that the entire new generation of individuals is an exact copy of the previous population, except for those created through mutation.

In this thesis, CR of 0.2, 0.4, 0.6 and 0.8 are experimented. The effect of CR can be seen from Figure 5.4 below.

The value of 0.2 was not included in systematic tests because it did not converge quickly and occasionally did not reach the optimal value during exploratory tests.

The mutation rate (MR) in GA specifies the fraction of chromosomes that experience mutation in every generation. The primary objective of mutation is to avoid the GA from getting stuck in local optima, but if the mutation rate is too high, it can result in the GA behaving more like a random search algorithm (Maschek, 2010). According to Schlierkamp-Voosen (1993), a mutation is more effective for small populations, and a mutation rate of 1/PS is considered efficient.

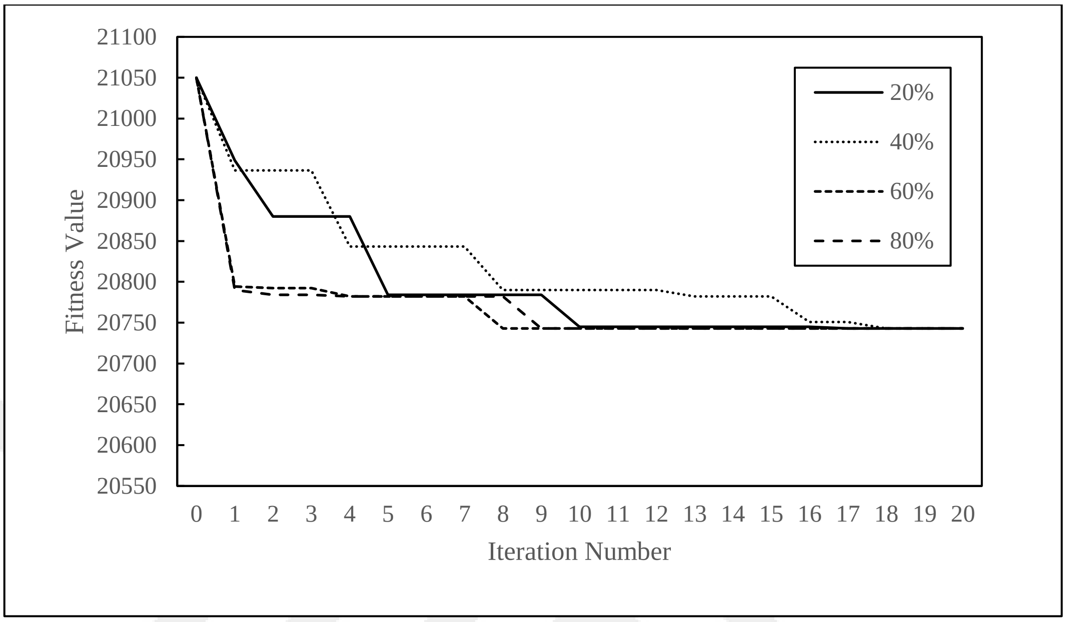

Figure 5.4 The effect of CR on convergence for SimNGA

Previous research has suggested different values for MR for various problems. In this thesis, experiments were conducted using MR values of 0.1, 0.2, 0.3 and 0.4. The effect of MR on the TrainSchPrb can be seen from Figure 5.5.

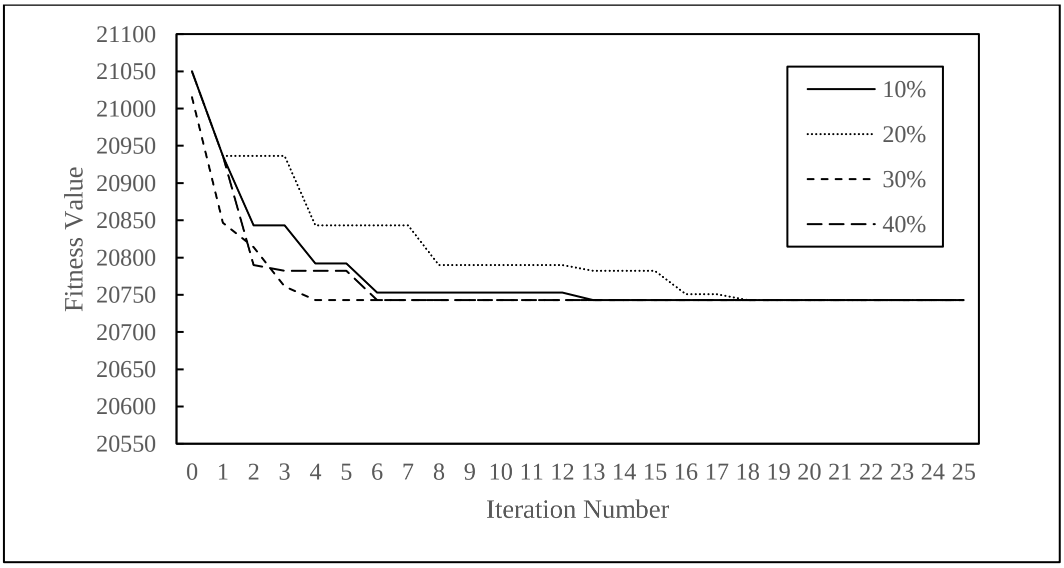

Figure 5.5 The effect of MR on convergence for SimNGA

After conducting exploratory tests, it is observed that the value of 0.4 converges quickly; however, it has shown performance below other values in reaching optimal results. Therefore, it has been deemed appropriate not to include this value in systematic tests.

The maximum number of generations is the number of cycles that occur before the GA terminates. The required number of cycles depends on the problem's complexity and type. Depending on the GA design, this parameter may not be used. In the TrainSchProb, SimNGA is terminated if the generation consistently produces identical best fitness value for a specified number of iterations. After numerous observations, it has been determined that 1000 is sufficient for this number.

The algorithm is executed for each unique combination, and the outcomes are presented in Appendix A.2. Furthermore, the top ten solutions generated by SimNGA are presented in Table 5.11.

As shown in the Table 5.11, the best average travel time value is 21693. Although SimNGA may not have achieved good results compared to SimGA and hybrid applications in terms of obtaining good results, it performs much better in terms of performance. Utilizing the speed of this algorithm can be beneficial in order to achieve better results.

Table 5.11 Top ten fitness values for SimNGA

Number	**Configuration**	**Fitness Value**	**PS**	**CR**	**MR**
1	6,3,4,2,2,4,5,6,4	21693	10	0.4	0.2
2	1,3,1,2,4,4,5,6,4	21698	10	0.6	0.3
3	1,3,1,2,4,4,4,6,5	21698	20	0.4	0.3
4	1,6,1,2,2,5,4,1,4	21702	10	0.4	0.3
5	1,3,4,2,4,1,2,1,4	21705	10	0.8	0.3
6	1,3,1,2,2,4,2,1,4	21706	10	0.8	0.1
7	1,3,2,2,4,4,4,6,5	21706	20	0.4	0.1
8	6,3,1,2,4,4,5,6,4	21708	10	0.6	0.1
9	1,3,4,2,4,4,2,3,5	21708	20	0.4	0.2
10	1,3,3,2,4,4,2,3,5	21710	20	0.6	0.2

5.3 SimSA for the TrainSchProb

Simulated Annealing (SA) is a versatile algorithm that can be used to solve many problems related to combinatorial and function optimization. Its flexibility makes it a suitable choice for various problems. SA involves setting various parameters such as the initial temperature, the number of perturbations to be made at each temperature level, the cooling rate, and the stopping criterion. Additionally, SA may require modifications to the algorithm and the representation of the problem to be optimized.

A heuristic SA algorithm is proposed to solve the model, which is effective for solving large combinatorial problems and can produce satisfactory solutions within a reasonable timeframe. The framework of SA is given in Table 5.12.

Table 5.12 Framework of SimSA

```
Start SimSA
GENERATE initial solution x^c
g_i: decision point
x^c = [g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, g_9],
        For i = 1 To 9
                g_i: takes a random value from set N: {1,2,3,4,5,6}
        Next i
INITIALIZE R_max, E and α
        WHILE E < E_0 DO
                FOR r = 1 to R_max DO
                        x^n= adjacent solution in NS (all possible neighboring solutions)
                        Generate x^n ϵ NS
                        Compute Δ= f(x^n) − f(x^c) and generate u
                        IF (Δ< 0) or exp(e^(−Δ/T)) > u THEN x^c = x^n
                Next r
                REDUCE E
        END WHILE
End SimSA
```

As shown in Table 5.12, SA operates with inputs including the temperature E, the minimum temperature E_0, and cooling rate α. The algorithm loops over the temperature E, progressively decreasing it until it reaches a value close to E_0. During this loop, at each iteration, an adjacent candidate solution is created by modifying the current solution (x^{c}). This change leads to a neighboring solution (x^{n}), selected randomly. The newly generated neighboring solution is accepted and replaces the current one based on a probability distribution.

5.3.1 Elements of SimSA

In this subsection elements of SimNGA are presented.

5.3.1.1 Solution Representation

A suitable representation of the control variables is typically the most apparent choice when applying the SA algorithm to solve an optimization problem. The solution representation used in SimSA algorithm is similar to SimNGA algorithm, consisting of 9 decision variables where each decision variable can take values 1, 2, 3, 4, 5, and 6 corresponding to the queue rule for tracks between each station.

- g_i: decision point from set N = {1,2,3,4,5,6}
- $x^c = [g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, g_9]$; current solution

5.3.1.2 Solution Generation

A careful consideration is required for how new solutions are produced. Busetti, F. (2003) stated that one should design the solution generator in a way that it:

- makes small random alterations, and
- enables the generation of all possible solutions.

In the case of combinatorial or permutation optimization problems, the way solutions are represented and generated is specific to the problem at hand. For the proposed SimSA method, random decision point will be selected, and its value is changed in order to find a neighboring solution.

5.3.1.3 Cooling Schedule

When choosing a suitable cooling schedule for the SA algorithm, it is important to set the initial temperature at a high enough level to fully melt the system, and then gradually lower it towards the freezing point as the search progresses. Additionally, to implement the SA algorithm following parameters is required:

- E: initial temperature
- α: cooling rate
- E_0: final temperature

- A rule for decrementing temperature

In this thesis, considering the challenges of TrainSchPrb and previous research, the SimSA is subjected to multiple computational experiments to assess the effectiveness of the proposed solution method.

5.3.2 Parameter Tuning and Computational Results for SimSA

First critical parameter of SimSA is initial temperature E. It should be high enough to allow exploration in the search space.

As seen from the Figure 5.6, the results converge to the same value for all tested parameters. Having an initial temperature higher than 1000 requires much more computational time without providing any improvement, so it should not be preferred.

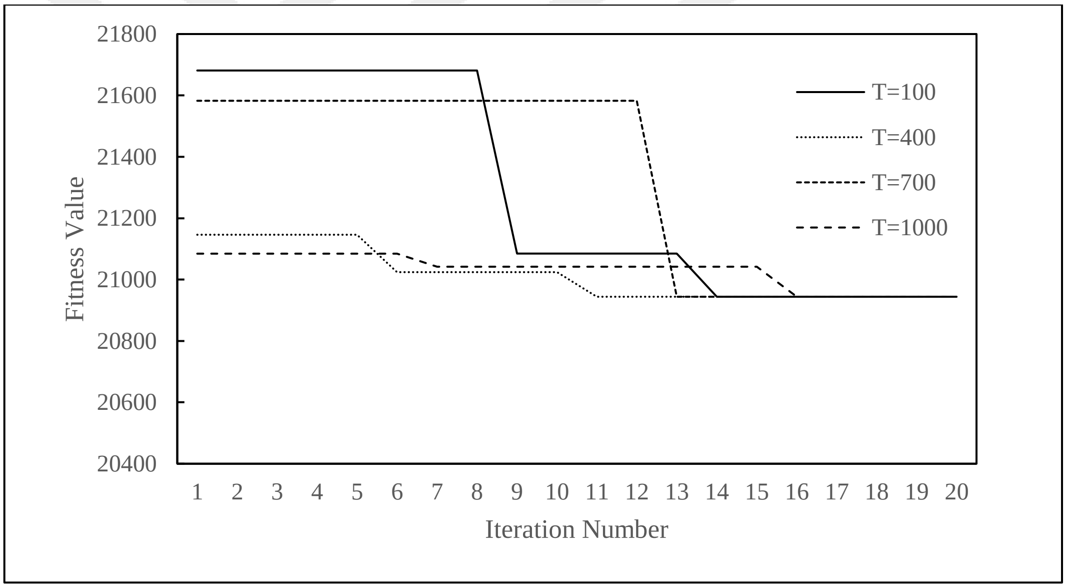

Figure 5.6 The effect of initial temperature on convergence for SimSA

Second parameter to consider for SimSA is cooling rate α which determines how fast the temperature decreases during the process.

As can be seen from the Figure 5.7, among the tested values, 0.95, 0.97, and 0.99 converge faster than the others. Therefore, it has been decided to use these values in the systematic testing phase.

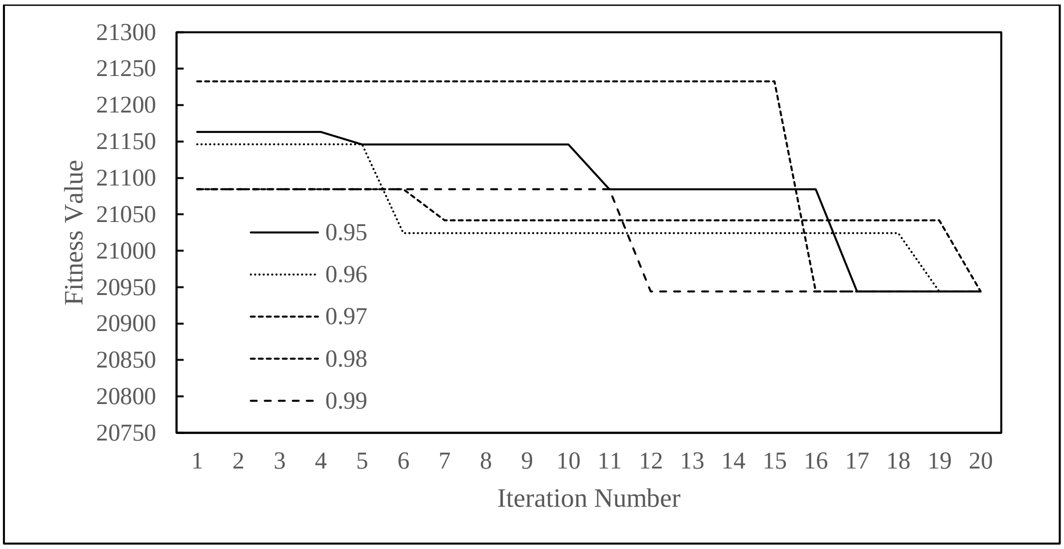

Figure 5.7 The effect of α on convergence for SimSA

Another important parameter is R_{max}. As seen from the Figure 5.8, values lower than 500 do not lead to the desired outcome. However, if this parameter exceeds 1500, the solution time becomes excessively long, making it impractical for real-world use. Therefore, for systematic testing, values of 500, 1000, and 1500 have been deemed appropriate.

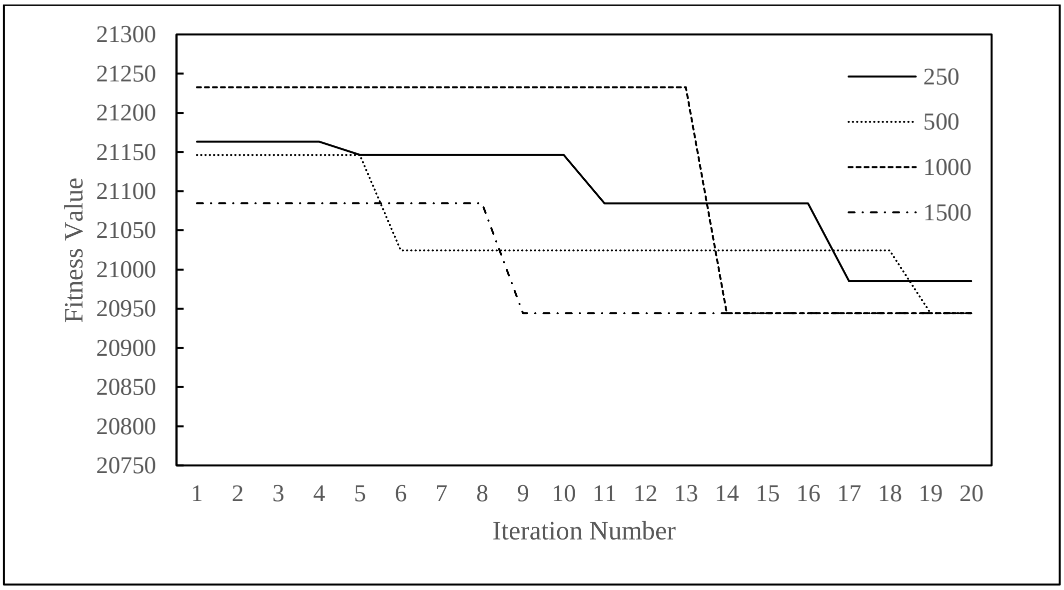

Figure 5.8 The effect of R_{max} on convergence for SimSA

Lastly, the parameter tested is final temperature E_0 which determines when the algorithm will be terminated.

From the Figure 5.9, it can be seen that values of 0.01 and lower can be set in order to get desired results.

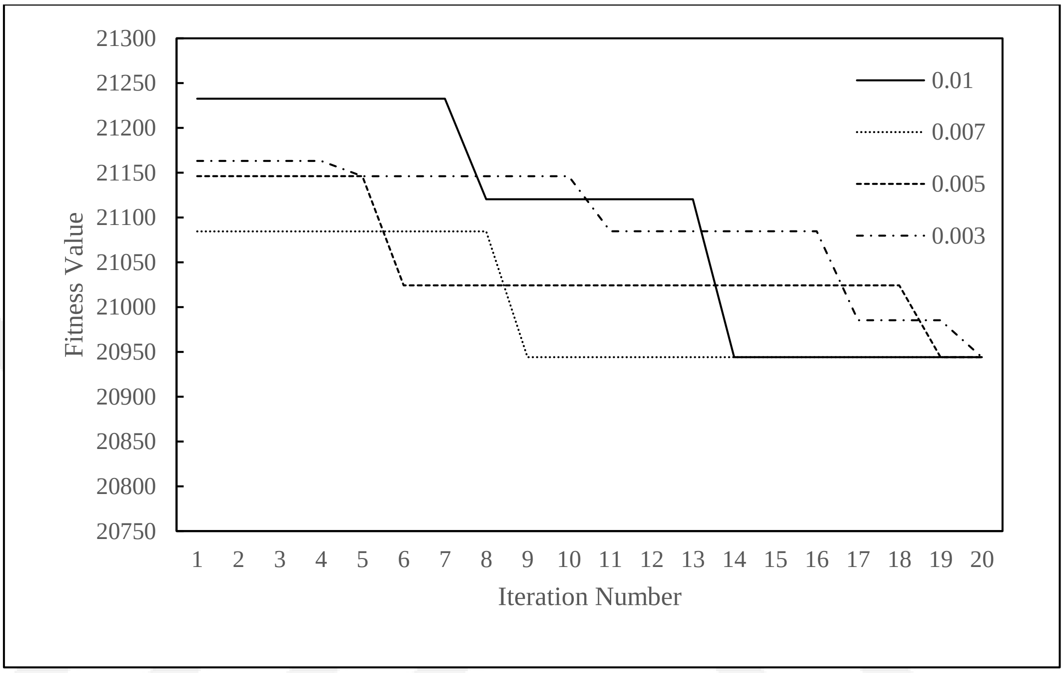

Figure 5.9 The effect of final temperature on convergence for SimSA

The algorithm is run for each different combination, and the outcomes are presented in Appendix A.3. Furthermore, the top ten solutions generated by SimSA are presented in Table 5.13.

Table 5.13 Top ten fitness values for SimSA

Number	Configuration	Fitness Value	E_0	α	N	E
1	5,6,4,2,4,1,5,1,4	21704	500	0.99	1500	0.007
2	5,3,5,2,4,4,5,1,4	21704	300	0.95	1500	0.005
3	5,3,6,3,4,1,4,3,4	21711	100	0.97	1500	0.01
4	5,3,6,3,4,1,4,3,4	21711	300	0.95	1000	0.007
5	5,6,1,2,4,3,5,1,4	21722	100	0.95	1000	0.007
6	5,6,1,2,4,3,5,1,4	21722	100	0.95	1000	0.01
7	6,3,2,2,4,5,6,5,5	21723	300	0.95	1000	0.005
8	6,3,2,2,4,5,6,5,5	21723	500	0.95	500	0.01
9	6,3,2,2,4,5,6,5,5	21723	500	0.99	500	0.005
10	6,5,1,6,4,4,5,4,4	21744	1000	0.99	1500	0.01

As shown in the Table 5.13, the best average travel time value is 21704. It can be easily inferred that SimSA alone is not sufficient for TrainSchProb. Therefore, it has been deemed not beneficial to use it in the hybridization process.

5.4 SimTS for the TrainSchProb

TS is an iterative method that creates an extended neighborhood while focusing on preventing being stuck in a local optimal solution. The method involves examining the search space by transitioning from a solution to its best neighbor, even if this leads to a decrease in the objective function value. This approach increases the probability of moving away from local optima. In order to take advantage of this property, a TS algorithm is proposed for the TrainSchProb. The framework of SimTS is depicted in Table 5.14.

Table 5.14 Framework of SimTS

```
Start SimTS
GENERATE initial solution s_0
g_i: decision point
s_0 = [g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, g_9]
        For i = 1 To 9
                g_i: takes a random value from set N: {1,2,3,4,5,6}
        Next i
INITIALIZE aspiration criteria and termination criteria
tabu list size (TL) = {}
s_best (best solution) = s_0 = s (current solution)
        WHILE stopping criteria is not met DO
                Generate s
                IF s ∉ Tl
                        IF fitness(s) > fitness(s_best)
                            s_best = s
                        END
                END
                UPDATE tabu list and aspiration criteria
        END
Stop SimTS
```

5.4.1 Parameter Tuning and Computational Results for SimTS

In this subsection, parameter tuning and computational results for SimTS are given.

The first observed parameter of SimTS is the maximum number of non-improvement iterations (maxNIter). This parameter sets the upper limit for the number of iterations or search steps permitted in the SimTS.

As seen in Figure 5.10, the tests conducted on the maxNIter parameter show that as the value increases, the desired value is reached more quickly. However, if this parameter exceeds 2000, it significantly extends the computational time, rendering it impractical for real-world applications. As a result, values of 1000, 1500, and 2000 have been selected.

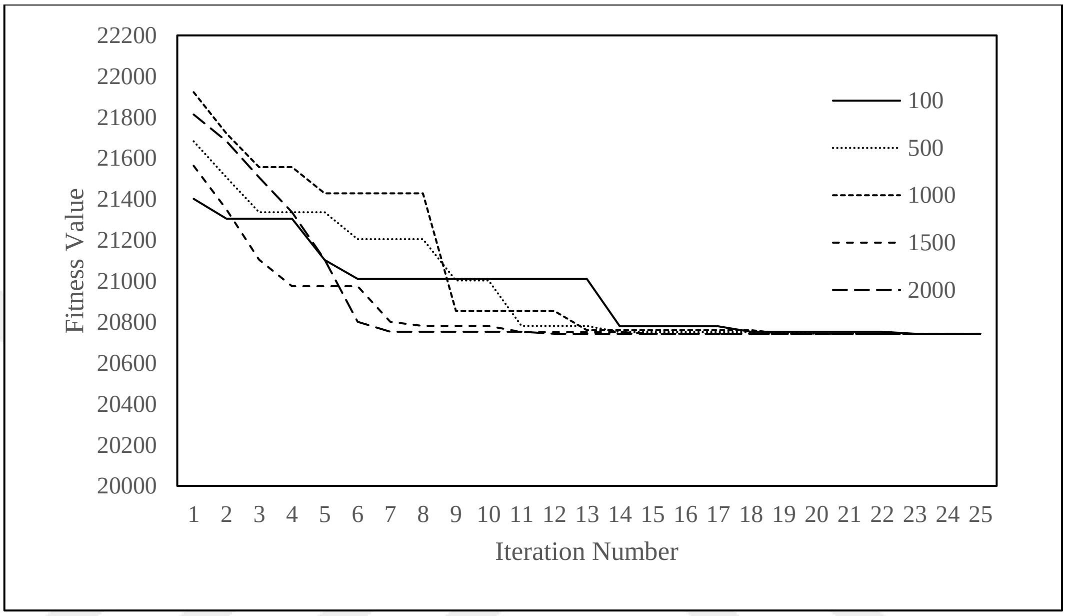

Figure 5.10 The effect of maxNIter on convergence for SimTS

Another important parameter for SimTS is neighborhood size which is the number of neighboring solutions explored in each iteration.

The exploratory tests indicate that as the neighborhood size increases, the desired solution is reached more rapidly. However, after the value of 15, the algorithm becomes less desirable as the computational time increases significantly. Hence, values of 5, 10, and 15 are used for systematic testing.

Finally, exploratory tests are conducted for the tabu tenure parameter, which determines the length of time during which a move is considered tabu.

Figure 5.11 and Figure 5.12 shows the effect of neighborhood size and tabu tenure parameters.

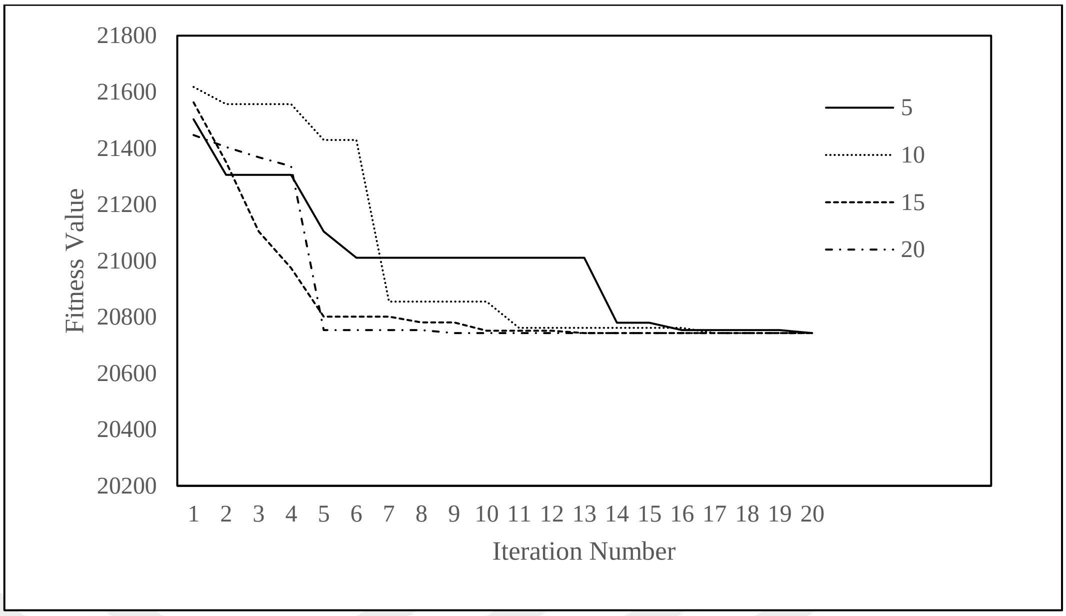

Figure 5.11 The effect of neighborhood size on convergence for SimTS

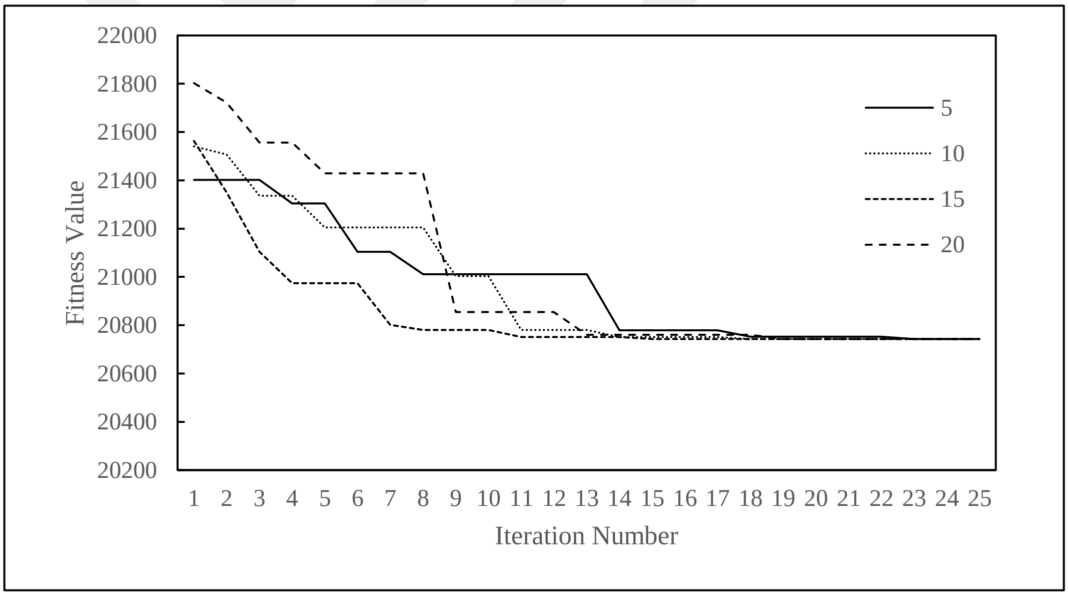

Figure 5.12 The effect of tabu tenure on convergence for SimTS

Test results showed that smaller tabu tenure improves algorithm’s performance without decreasing its quality. That is why, maximum value of the tabu tenure is set to 20.

As displayed in the Table 5.15, the best average travel time value is 21679. With SimTS, a better solution is found with respect to SimNGA and SimSA. Also, the top ten solutions generated by SimNGA are presented in Table 5.16.

Table 5.15 Computational results for SimTS

			maxNIter=100		maxNIter=1000		maxNIter=2000	
	Tabu Tenure	**Neighbor Number**	**Solution**	**Fitness**	**Solution**	**Fitness**	**Solution**	**Fitness**
1	5	5	6,3,1,2,5,5,4,6,4	21787	3,3,4,2,5,5,2,6,5	21782	1,3,5,2,2,1,5,3,5	21683
2	5	10	6,3,4,2,2,1,5,1,5	21690	1,6,1,2,2,1,2,6,5	21724	1,3,1,2,2,4,2,1,5	21706
3	5	15	3,6,1,2,4,1,2,6,5	21727	1,6,1,2,2,1,4,3,5	21721	3,6,1,2,4,1,2,6,5	21727
4	10	5	1,3,1,2,4,1,5,3,5	21714	1,6,4,2,2,4,4,1,4	21679	1,6,5,2,2,4,2,3,4	21693
5	10	10	1,6,1,2,4,4,2,1,5	21702	3,6,4,2,2,4,4,1,4	21689	6,3,1,2,4,5,4,6,4	21709
6	10	15	1,3,1,2,4,4,5,3,5	21698	1,3,1,2,4,1,4,1,5	21711	3,2,5,2,4,1,5,3,5	21729
7	15	5	1,6,4,2,2,4,2,1,4	21690	1,6,1,2,5,4,4,6,5	21776	1,6,5,2,2,4,2,3,5	21693
8	15	10	3,6,4,2,2,4,4,1,4	21689	1,6,4,2,5,1,5,1,5	21774	3,6,1,2,2,1,2,6,5	21733
9	15	15	3,3,5,2,2,5,4,3,5	21694	1,6,4,2,2,5,4,1,5	21680	1,6,5,2,5,4,5,6,5	21775
10	20	5	1,3,1,2,2,4,4,6,5	21703	1,6,5,2,2,4,2,3,5	21693	3,3,4,2,2,1,5,1,5	21690
11	20	10	6,6,5,2,4,5,2,1,5	21722	1,3,1,2,4,1,4,1,5	21711	1,6,1,2,4,1,2,6,5	21718
12	20	15	1,6,1,2,2,4,2,1,5	21706	1,3,1,2,4,1,5,3,5	21714	3,3,4,2,5,5,2,6,5	21782

Table 5.16 Top ten fitness values for SimTS

Number	Configuration	Fitness Value	Tabu tenure	Neighbor number	maxIter
1	1,6,4,2,2,4,4,1,4	21679	10	5	1000
2	1,6,4,2,2,5,4,1,5	21680	15	15	1000
3	1,3,5,2,2,1,5,3,5	21683	5	5	2000
4	3,6,4,2,2,4,4,1,4	21689	10	10	1000
5	3,6,4,2,2,4,4,1,4	21689	15	10	100
6	6,3,4,2,2,1,5,1,5	21690	5	10	100
7	1,6,5,2,2,4,2,3,5	21693	20	5	1000
8	1,6,5,2,2,4,2,3,4	21693	10	5	2000
9	1,6,5,2,2,4,2,3,5	21693	15	5	2000
10	3,3,5,2,2,5,4,3,5	21694	15	15	100

As it can be seen from the Table 5.16, SimTS has provided very good results compared to other algorithms. Therefore, due to its promising results and performance, it is considered appropriate to utilize SimTS in the hybridization process.

5.5 Hybrid TS-GA (SimHA) for TrainSchProb

In this subsection, a hybrid metaheuristic approach that combines GA with the TS is presented. In essence, hybrid algorithm primarily utilizes the GA as its main algorithm, while also incorporating the TS procedure to enhance individuals within the population.

5.5.1 Hybridization of SimNGA with Tabu Search (TS)

To create a hybrid algorithm, one of the following approaches can be used:

- Substitute the mutation operator in the GA with the TS algorithm. Rather than randomly mutating a gene, use the TS algorithm to generate a novel solution that is not present in the tabu list.
- Use the GA to generate an initial population of solutions, and then apply the TS algorithm to each solution in the population. This can be done for a fixed number of iterations, or until a termination condition is met.
- Use the GA to generate a set of promising solutions, and then use the TS algorithm to refine each solution. This can be done iteratively until convergence is reached.

The selection of a suitable hybrid GA and TS algorithm approach depends on the problem characteristics and the optimization task requirements. If the problem has a large search space or a complex fitness landscape, it may be advantageous to use the GA to generate an initial set of solutions and then refine each solution using the TS algorithm. This strategy can prevent local optima and can enhance the quality of the solutions. Since TrainSchProb is complex and has a risk of getting stuck in local optima, third option will be utilized.

Tabu search (TS) is a meta-heuristic technique that has demonstrated effectiveness in solving a variety of combinatorial optimization problems, including various scheduling problems. Tabu search involves multiple components, including the structure of the neighborhood, the attributes of the moves, the tabu list, criteria for aspiration, and criteria for termination. The procedure for TS is denoted in Table 5.17.

Additionally, SimHA framework for the hypothetical TrainSchPrb is presented in Table 5.18.

Table 5.17 TS procedure

Start Tabu Search Algorithm
Set best solution (S_{best}) = current solution S , (TL) = {}
For i = 1 To maximum iteration
For j = 1 To neighborhood size
Randomly generate new solution
If $S \notin TL$
If (fitness (S) > fitness (S_{best}))
S_{best}= S
End
End
Next j
Next i
Stop

Table 5.18 Framework of SimHA

Start SimHA
Define GA parameters
popSize: population size
mtR: mutation rate
crR: crossover rate
maxGen: maximum generation number
Define Tabu search algorithm parameters
negSize: neighborhood size
Tabu Tenure
maxIter: maximum number of iterations
Initialize population
genN: generation number
For genN = 1 To maxGen
Perform Crossover
Perform Mutation
Rank the population
For j = 1 To popSiz
Perform Tabu Search Algorithm
Next j
Record the optimal fitness score
Rank the population
Check termination criteria
Next genN
Stop SimHS

5.5.2 Parameter Tuning and Computational Results for SimHA

Similar parameter tuning procedures as in SimNGA, SimTSA, and SimSA algorithms were followed to calibrate the proposed hybrid algorithm. The exploratory testing begins with the components and parameters of GAs and continues with the TS algorithm, which is used for local search. After this step, parameters or components that are robust or significantly superior are fixed. Finally, systematic testing for other parameters or components are carried out.

Based on the tests performed for SimNGA, it was observed that the population size, mutation rate, and crossover rate played crucial roles in the subsequent systematic testing phase.

As shown in Figure 5.13, it is observed during the tests that the optimal result was achieved at each population size, but the solution converges to the optimal value most rapidly when the population size is set to 20. Additionally, the computational time becomes unacceptable after the value of 30. That is why, similar to SimNGA, the values of 10, 20, and 30 were selected.

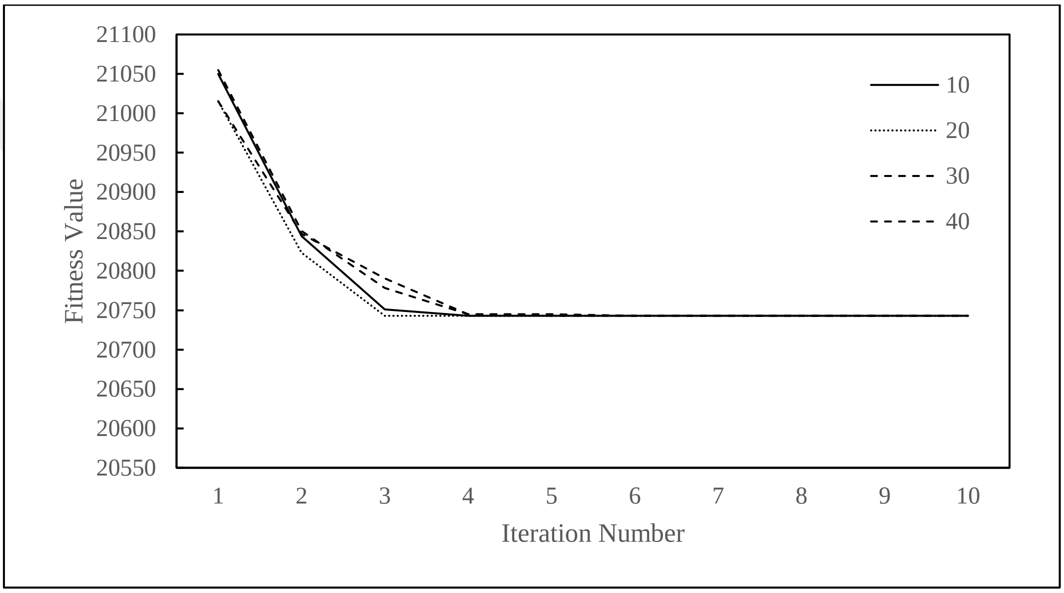

Figure 5.13 The effect of population size on convergence for SimHA

Due to the importance of testing CR and MR parameters over a wide range of values, it was decided to use the same values as in the SimNGA for the systematic tests of the SimHA algorithm.

Given the integration of SimNGA with the Tabu Search algorithm, it became crucial to examine the impact of parameters such as the maximum non-improvement iteration number, neighborhood size, and tabu tenure on the solution.

As shown in Figure 5.14, it was observed during the tests that the optimal result was achieved at each maxNIter, but the population converged to the optimal value most rapidly when the maximum iteration number was set to 200. Also, values exceeding 200 were not efficient in terms of computational time. Therefore, values of 100, 150, and 200 were chosen for systematic testing.

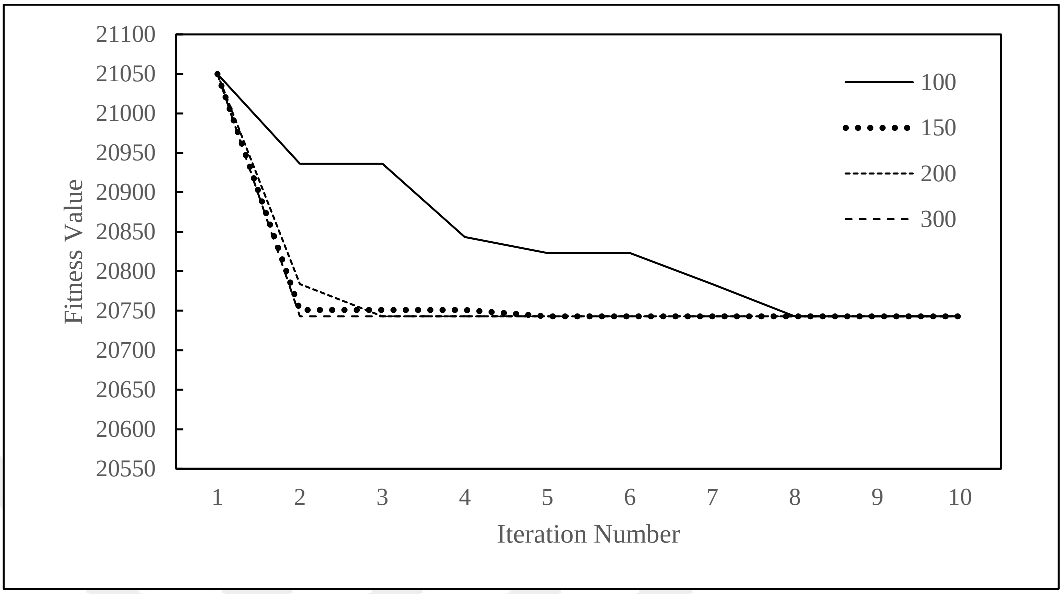

Figure 5.14 The effect of maxNIter on convergence for SimHA

During the tests conducted for the neighbor size parameter, it was observed that values less than 5 occasionally fail to achieve the best result, and for larger values, although the algorithm converges to the result quickly, the running time becomes unacceptably long. Therefore, it is deemed appropriate to use a value of 5 for the neighbor size parameter in the tests of the SimHA algorithm.

Finally, exploratory tests conducted on the tabu tenure parameter showed that SimHA is quite robust for different values of this parameter. Therefore, a value of 5 is selected for the tabu tenure parameter. Figure 5.15 and Figure 5.16 shows the impact of the neighborhood size and tabu tenure parameters.

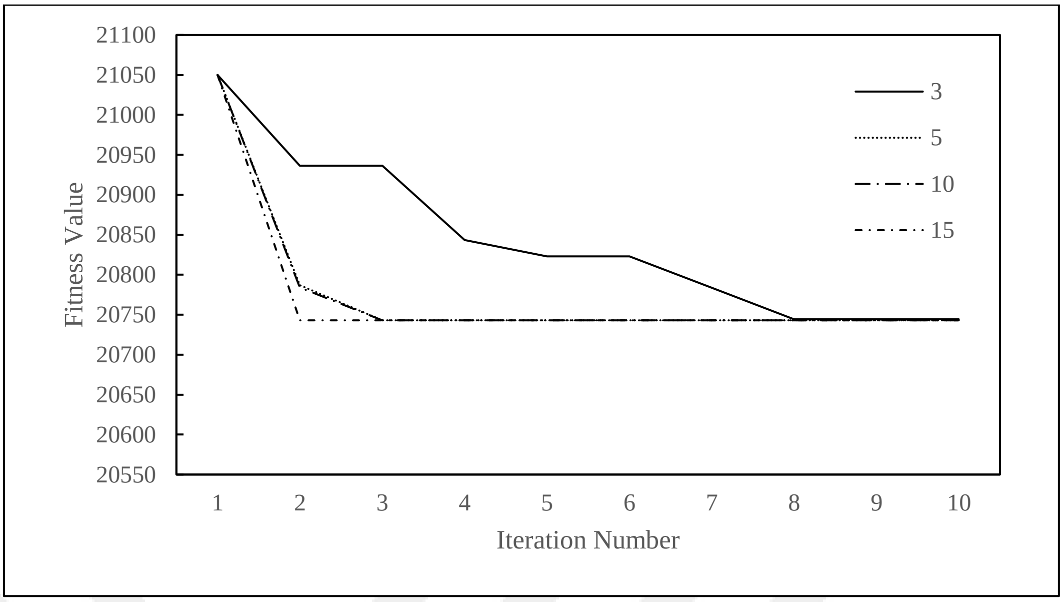

Figure 5.15 The effect of neighborhood size on convergence for SimHA

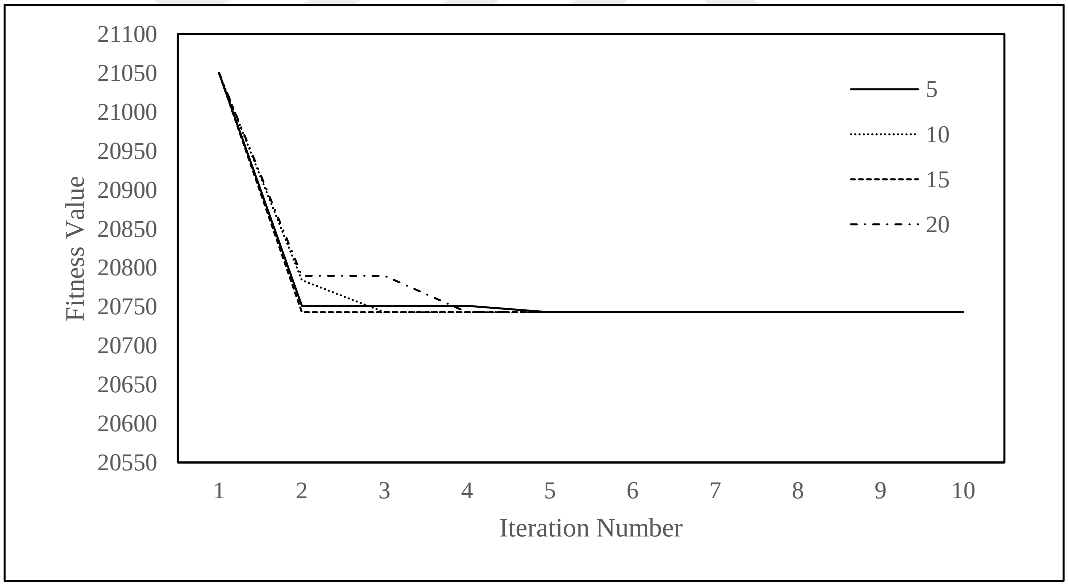

Figure 5.16 The effect of tabu tenure on convergence for SimHA

The algorithm is executed for each unique combination, and the outcomes are presented in Appendix A.4. In addition to that, top ten results of SimHA are given in Table 5.19.

Table 5.19 Top ten fitness values for SimHA

Number	Configuration	Fitness Value	PS	CR	MR	maxIter
1	1,3,1,2,4,4,2,6,4	21665	10	0.6	0.3	100
2	1,3,1,2,4,4,2,6,4	21665	20	0.4	0.3	100
3	1,3,1,2,4,4,2,6,4	21665	10	0.8	0.3	150
4	1,3,4,2,2,4,4,1,4	21679	20	0.6	0.1	100
5	1,3,4,2,2,4,4,1,4	21679	20	0.4	0.2	150
6	1,6,4,2,2,5,4,1,5	21680	30	0.6	0.3	100
7	1,3,1,2,4,4,2,1,5	21683	10	0.4	0.2	150
8	1,3,1,2,2,5,2,6,5	21688	20	0.6	0.3	150
9	1,3,1,2,2,5,2,6,5	21688	10	0.8	0.2	200
10	3,6,4,2,2,4,4,6,5	21693	30	0.4	0.2	150

As shown in the Table 5.19, the best average travel time value is 21665. This result is the best among those achieved so far.

After conducting the experiments, the top ten results obtained are presented in Table 5.20. The table includes the best configurations, fitness values, solution methods, and experiment numbers. As can be seen from the table, the SimHA algorithm has outperformed other solution approaches in achieving good results. In addition, the hybrid GAs proposed by Yalçınkaya (2010) and the SimTS algorithm have also achieved highly successful results.

Table 5.20 Top ten fitness values among all proposed approaches

Number	Configuration	Fitness Value	Solution Method	Experiment Number
1	1,3,1,2,4,4,2,6,4	21665	SimHA	6,12,36
2	1,3,1,2,2,5,4,1,5	21679	SimHA	34
3	1,6,4,2,2,4,4,1,4	21679	SimTS	16
4	1,6,4,2,2,5,4,1,5	21680	SimHA; SimTS	24; 21
5	1,3,1,2,4,4,2,1,5	21683	SimHA	29
6	1,3,1,2,2,5,2,6,5	21688	SimHA	42, 62
7	6,3,4,2,2,4,4,1,4	21689	SimGA; SimGAb; SimGAfs	31,36; 61,62; 88
8	3,6,4,2,2,4,4,1,4	21689	SimTS	8, 17
9	6,3,4,2,2,5,5,1,4	21690	SimGA	33
10	3,3,4,2,2,1,5,1,5	21690	SimTS	34

Among the top 10 results found, 9 were obtained by SimHA, 5 by SimTS, 3 by SimGA, 2 by SimGAb, and 1 by SimGAfs.

When we made a statistical analysis, as shown in Table 5.20, it is seen that there is no statistically significant difference between best result and 10th best result. However, further analysis showed significant difference for the many other results.

Table 5.21 Output analysis between 1st and 10th best results

Paired- Means Comparison: Number1-Number10

IDENTIFIER	Estd. Mean Difference	Standard Deviation	0.950 C.I. Half-Width	Minimum Value	Maximum Value	Number of Observations
AverageTravel	12.6	632	55.5	1.8e+004	3.873+004	500
TimeofTrains				1.8e+004	3.873+004	500
	Fail to Reject H_0 => Means are equal at 0.05 level					

CHAPTER SIX
CONCLUSION

In this thesis, we embarked on a comprehensive exploration of train scheduling, a critical aspect of modern rail systems' efficiency and reliability. Our investigation aimed to address the challenges posed by increasing traffic, the demand for enhanced service quality, and the evolving landscape of railway management. Through extensive research and experimentation, we have made significant contributions to the field of train scheduling optimization.

First and foremost, our review of the literature highlighted the historical evolution of train scheduling methods and underscored the need for innovative approaches. As rail systems are transformed into more liberalized and privatized entities, the emphasis on competitiveness and profit orientation intensifies. Moreover, the growing ecological concerns and governmental policies aimed at shifting transportation from roads to railways underscore the strategic importance of efficient train scheduling.

To tackle these multifaceted challenges, we proposed a simulation-integrated optimization framework. By combining metaheuristic algorithms with simulation modeling, we created a versatile approach capable of addressing the stochastic and dynamic nature of real-world rail systems. Our findings confirmed that this hybrid approach significantly improves scheduling efficiency while maintaining solution quality.

Furthermore, we conducted an in-depth analysis of various metaheuristic algorithms, including GA, SA, and TS. This exploration allowed us to identify the strengths and weaknesses of each method and laid the foundation for hybridization strategies. These hybrid approaches demonstrated exceptional potential for overcoming the TrainSchProb's complexities.

Our research also highlighted the pivotal role of parameter tuning. Factors such as population size, mutation rate, and crossover rate were identified as key influencers of algorithm performance. By fine-tuning these parameters, we achieved remarkable improvements in solution quality and computation time.

In conclusion, this thesis advances the understanding of train scheduling optimization by introducing a simulation-integrated approach and comprehensively evaluating metaheuristic algorithms. We have contributed valuable insights to the field, paving the way for more efficient and reliable railway systems. As railway transportation continues to play a crucial role in sustainable mobility, our research serves as a foundation for further advancements in this critical domain.

The findings presented in this thesis open up opportunities for future research into advanced optimization techniques, real-world implementations, and the integration of emerging technologies to further revolutionize train scheduling and, by extension, modern rail systems.

Although this thesis has made significant strides in addressing train scheduling problem, there are inherent limitations that should be recognized.

- Studied simulation model, while comprehensive, makes certain simplifications of real-world factors such as unexpected weather conditions. Incorporating these elements into the model could provide a more accurate representation of the challenges faced by real-world railway systems.
- The performance of our metaheuristic algorithms is contingent on specific parameter values. Sensitivity to these values has been observed, and a more in-depth analysis of parameter tuning and its impact on solution robustness could improve algorithm performance.
- The computational requirements of our algorithms might pose challenges in terms of time and resource efficiency. Future research could explore ways to optimize these algorithms further to expedite the optimization process.

Building upon the current work, several avenues for future research emerge:

- Exploring the integration of machine learning techniques, such as reinforcement learning, could enable our model to adapt and learn from dynamic changes in the railway environment.
- Extending the optimization framework to consider multiple conflicting objectives, such as minimizing waiting times and maximizing resource utilization simultaneously, would provide a more comprehensive solution that aligns with the complex nature of railway scheduling.
- Conducting field trials and implementing the developed algorithms in collaboration with railway authorities could validate the effectiveness of our

models in real-world scenarios. This step is crucial for ensuring the practical applicability and robustness of the proposed solutions.

Addressing these limitations and exploring these future research directions will contribute to the continuous improvement and applicability of our proposed solutions in the realm of train scheduling optimization.

REFERENCES

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

Abels, D., Jordi, J., Ostrowski, M., Schaub, T., Toletti, A., & Wanko, P. (2021). Train scheduling with hybrid answer set programming. *Theory and Practice of Logic Programming*, *21*(3), 317-347.

Acuna-Agost, R., Michelon, P., Feillet, D., & Gueye, S. (2011). A MIP-based local search method for the railway rescheduling problem. Networks, 57(1), 69-86.

Acuna-Agost, R., Michelon, P., Feillet, D., & Gueye, S. (2011). SAPI: Statistical Analysis of Propagation of Incidents. A new approach for rescheduling trains after disruptions.

Almodóvar, M., & García-Ródenas, R. (2013). On-line reschedule optimization for passenger railways in case of emergencies. *Computers & Operations Research*, *40*(3), 725-736.

Burdett, R. L., & Kozan, E. (2010). A disjunctive graph model and framework for constructing new train schedules. *European Journal of Operational Research*, *200*(1), 85-98.

Cacchiani, V., Furini, F., & Kidd, M. P. (2016). Approaches to a real-world train timetabling problem in a railway node. Omega, 58, 97-110.

Cacchiani, V., Huisman, D., Kidd, M., Kroon, L., Toth, P., Veelenturf, L., & Wagenaar, J. (2014). An overview of recovery models and algorithms for real-time railway rescheduling. *Transportation Research Part B: Methodological*, *63*, 15-37.

Cacchiani, V., & Toth, P. (2012). Nominal and robust train timetabling problems. *European Journal of Operational Research*, *219*(3), 727-737.

Cai, X., & Goh, C. J. (1994). A fast heuristic for the train scheduling problem. *Computers & Operations Research*, *21*(5), 499-510.

Canca, D., Zarzo, A., Algaba, E., & Barrena, E. (2011). Confrontation of different objectives in the determination of train scheduling. *Procedia-Social and Behavioral Sciences*, *20*, 302-312.

Crainic, T. G., Gendreau, M., Soriano, P., & Toulouse, M. (1993). A tabu search procedure for multicommodity location/allocation with balancing requirements. *Annals of Operations research*, *41*, 359-383.

Corman, F., D'Ariano, A., Marra, A. D., Pacciarelli, D., & Samà, M. (2017). Integrating train scheduling and delay management in real-time railway traffic control. Transportation Research Part E: Logistics and Transportation Review, 105, 213-239.

Corman, F., D'Ariano, A., Pacciarelli, D., & Pranzo, M. (2010). Centralized versus distributed systems to reschedule trains in two dispatching areas. *Public Transport*, *2*(3), 219-247.

Dollevoet, T., Corman, F., D'Ariano, A., & Huisman, D. (2014). An iterative optimization framework for delay management and train scheduling. *Flexible Services and Manufacturing Journal*, *26*(4), 490-515.

Espinosa-Aranda, J. L., García-Ródenas, R., Cadarso, L., & Marín, Á. (2014). Train scheduling and rolling stock assignment in high speed trains. Procedia-Social and Behavioral Sciences, 160, 45-54.

Fang, W., Yang, S., & Yao, X. (2015). A survey on problem models and solution approaches to rescheduling in railway networks. IEEE Transactions on Intelligent Transportation Systems, 16(6), 2997-3016.

Fischetti, M., & Monaci, M. (2017). Using a general-purpose mixed-integer linear programming solver for the practical solution of real-time train rescheduling. European Journal of Operational Research, 263(1), 258-264.

Frank, O. (1966). Two-way traffic on a single line of railway. *Operations Research*, *14*(5), 801-811.

Garrisi, G., & Cervelló-Pastor, C. (2020). Train-Scheduling Optimization Model for Railway Networks with Multiplatform Stations. Sustainability, 12(1), 257.

Gen, M., Cheng, R., & Lin, L. (2008). Network models and optimization: Multiobjective genetic algorithm approach. London: Springer.

Glover, F. (1989). Tabu search—part I. ORSA Journal on computing, 1(3), 190-206.

Gong, C., Shi, J., Wang, Y., Zhou, H., Yang, L., Chen, D., & Pan, H. (2021). Train timetabling with dynamic and random passenger demand: A stochastic optimization method. Transportation Research Part C: Emerging Technologies, 123, 102963.

Gotshall, S., & Rylander, B. (2002). Optimal population size and the genetic algorithm. *Population*, *100*(400), 900.

Hassanat, A., Almohammadi, K., Alkafaween, E. A., Abunawas, E., Hammouri, A., & Prasath, V. S. (2019). Choosing mutation and crossover ratios for genetic algorithms—a review with a new dynamic approach. *Information*, *10*(12), 390.

Hassannayebi, E., Boroun, M., Jordehi, S. A., & Kor, H. (2019). Train schedule optimization in a high-speed railway system using a hybrid simulation and meta-model approach. Computers & Industrial Engineering, 138, 106110.

Haupt, R. L., & Haupt, S. E. (2004). *Practical genetic algorithms*. John Wiley & Sons.

Haupt, R. L., & Haupt, S. E. (2004). *Practical genetic algorithms*. John Wiley & Sons.

Holland, J. H. (1992). *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press.

Huang, Y., Yang, L., Tang, T., Cao, F., & Gao, Z. (2016). Saving energy and improving service quality: Bicriteria train scheduling in urban rail transit systems. IEEE Transactions on Intelligent Transportation Systems, 17(12), 3364-3379.

Hu, H., Li, K., & Xu, X. (2013). A multi-objective train-scheduling optimization model considering locomotive assignment and segment emission constraints for energy saving. *Journal of Modern Transportation*, *21*(1), 9-16.

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

Jiang, Y., & Zhou, X. (2018). A connecting timetable rescheduling model for production and rail transportation with unexpected disruptions. IEEE Access, 7, 4284-4294.

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

Kirkpatrick, M. (1982). Quantum evolution and punctuated equilibria in continuous genetic characters. *The American Naturalist*, *119*(6), 833-848.

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

Kumar, A. (2013). Encoding schemes in genetic algorithm. *International Journal of Advanced Research in IT and Engineering*, *2*(3), 1-7.

Lamorgese, L., Mannino, C., & Natvig, E. (2017). An exact micro–macro approach to cyclic and non-cyclic train timetabling. Omega, 72, 59-70.

Liu, R., Li, S., Yang, L., & Yin, J. (2018). Energy-efficient subway train scheduling design with time-dependent demand based on an approximate dynamic programming approach. IEEE Transactions on Systems, Man, and Cybernetics: Systems.

Liu, S. Q., & Kozan, E. (2011). Scheduling trains with priorities: a no-wait blocking parallel-machine job-shop scheduling model. *Transportation Science*, *45*(2), 175-198

Li, X., & Lo, H. K. (2014). Energy minimization in dynamic train scheduling and control for metro rail operations. Transportation Research Part B: Methodological, 70, 269-284.

Li, X., Shou, B., & Ralescu, D. (2014). Train rescheduling with stochastic recovery time: A new track-backup approach. *IEEE Transactions on systems, man, and cybernetics: systems*, *44*(9), 1216-1233.

Maschek, M. K. (2010). Intelligent mutation rate control in an economic application of genetic algorithms. *Computational Economics*, *35*, 25-49.

Meng, L., & Zhou, X. (2014). Simultaneous train rerouting and rescheduling on an N-track network: A model reformulation with network-based cumulative flow variables. *Transportation Research Part B: Methodological*, *67*, 208-234.

Meng, X., Jia, L., & Qin, Y. (2010). Train timetable optimizing and rescheduling based on improved particle swarm algorithm. *Transportation Research Record*, *2197*(1), 71-79.

Mu, S., & Dessouky, M. (2011). Scheduling freight trains traveling on complex networks. *Transportation Research Part B: Methodological*, *45*(7), 1103-1123.

Narayanaswami, S., & Rangaraj, N. (2011). Scheduling and rescheduling of railway operations: A review and expository analysis. *Technology Operation Management*, *2*(2), 102-122.

Qi, J., Yang, L., Gao, Y., Li, S., & Gao, Z. (2016). Integrated multi-track station layout design and train scheduling models on railway corridors. Transportation Research Part C: Emerging Technologies, 69, 91-119.

Roeva, O., & Michalikova, A. (2013). Generalized net model of intuitionistic fuzzy logic control of genetic algorithm parameters. *Notes on Intuitionistic Fuzzy Sets*, *19*(2), 71-76.

Rutenbar, R. A. (1989). Simulated annealing algorithms: An overview. *IEEE Circuits and Devices magazine*, *5*(1), 19-26.

Sato, K., Tamura, K., & Tomii, N. (2013). A MIP-based timetable rescheduling formulation and algorithm minimizing further inconvenience to passengers. *Journal of Rail Transport Planning & Management*, *3*(3), 38-53.

Schlierkamp-Voosen, D. (1993, July). Optimal interaction of mutation and crossover in the breeder genetic algorithm. In *International Conference on Genetic Algorithms; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA*.

Shakibayifar, M., Sheikholeslami, A., & Corman, F. (2018). A simulation-based optimization approach to reschedule train traffic in uncertain conditions during disruptions. Scientia Iranica, 25(2), 646-662.

Shakibayifar, M., Sheikholeslami, A., & Jamili, A. (2018). A Multi-Objective Decision Support System for Real-Time Train Rescheduling. IEEE Intelligent Transportation Systems Magazine, 10(3), 94-109.

Shi, J., Yang, L., Yang, J., & Gao, Z. (2018). Service-oriented train timetabling with collaborative passenger flow control on an oversaturated metro line: An integer linear optimization approach. Transportation Research Part B: Methodological, 110, 26-59.

Sotskov, Y. N., & Gholami, O. (2012). Shifting bottleneck algorithm for train scheduling in a single-track railway. IFAC Proceedings Volumes, 45(6), 87-92.

Umbarkar, A. J., & Sheth, P. D. (2015). Crossover operators in genetic algorithms: a review. *ICTACT journal on soft computing*, *6*(1).

Van Laarhoven, P. J., Aarts, E. H., van Laarhoven, P. J., & Aarts, E. H. (1987). *Simulated annealing* (pp. 7-15). Springer Netherlands.

Wang, Y., Wei, Y., Zhang, Q., Shi, H., & Shang, P. (2019). Scheduling overnight trains for improving both last and first train services in an urban subway network. Advances in Mechanical Engineering, 11(5), 1687814019848920.

Wang, Y., Zhao, K., D'Ariano, A., Niu, R., Li, S., & Luan, X. (2021). Real-time integrated train rescheduling and rolling stock circulation planning for a metro line under disruptions. *Transportation Research Part B: Methodological*, *152*, 87-117.

Wang, X., Lv, Y., Sun, H., Xu, G., Qu, Y., & Wu, J. (2023). A simulation-based metro train scheduling optimization incorporating multimodal coordination and flexible routing plans. *Transportation Research Part C: Emerging Technologies*, *146*, 103964.

Wen, C., Huang, P., Li, Z., Lessan, J., Fu, L., Jiang, C., & Xu, X. (2019). Train dispatching management with data-driven approaches: a comprehensive review and appraisal. IEEE Access, 7, 114547-114571.

YALÇINKAYA, Ö. (2010). A feasible timetable generator simulation modelling framework and simulation integrated genetic and hybrid genetic algorithms for train scheduling problem (Doctoral dissertation, DEÜ Fen Bilimleri Enstitüsü).

Yalçınkaya, Ö., & Bayhan, G. M. (2012). A feasible timetable generator simulation modelling framework for train scheduling problem. *Simulation Modelling Practice and Theory*, *20*(1), 124-141.

Yang, L., Gao, Z., & Li, K. (2010). Passenger train scheduling on a single-track or partially double-track railway with stochastic information. *Engineering Optimization*, *42*(11), 1003-1022.

Yang, L., Qi, J., Li, S., & Gao, Y. (2016). Collaborative optimization for train scheduling and train stop planning on high-speed railways. Omega, 64, 57-76.

Yang, X., Li, X., Gao, Z., Wang, H., & Tang, T. (2012). A cooperative scheduling model for timetable optimization in subway systems. *IEEE Transactions on Intelligent Transportation Systems*, *14*(1), 438-447.

Yang, X., Li, X., Ning, B., & Tang, T. (2015). An optimisation method for train scheduling with minimum energy consumption and travel time in metro rail systems. *Transportmetrica B: Transport Dynamics*, *3*(2), 79-98.

Yin, J., Yang, L., Tang, T., Gao, Z., & Ran, B. (2017). Dynamic passenger demand oriented metro train scheduling with energy-efficiency and waiting time minimization: Mixed-integer linear programming approaches. Transportation Research Part B: Methodological, 97, 182-213.

Ying, C. S., Chow, A. H., & Chin, K. S. (2020). An actor-critic deep reinforcement learning approach for metro train scheduling with rolling stock circulation under stochastic demand. Transportation Research Part B: Methodological, 140, 210-235.

Yu, X., & Gen, M. (2010). *Introduction to evolutionary algorithms*. Springer Science & Business Media.

Zhan, S., Kroon, L. G., Zhao, J., & Peng, Q. (2016). A rolling horizon approach to the high speed train rescheduling problem in case of a partial segment blockage. Transportation Research Part E: Logistics and Transportation Review, 95, 32-61.

Zhang, H., Li, S., Wang, Y., Yang, L., & Gao, Z. (2020). Collaborative real-time optimization strategy for train rescheduling and track emergency maintenance of high-speed railway: a Lagrangian relaxation-based decomposition algorithm. Omega, 102371.

Zhang, Q., Lusby, R. M., Shang, P., & Zhu, X. (2022). A heuristic approach to integrate train timetabling, platforming, and railway network maintenance scheduling decisions. *Transportation Research Part B: Methodological*, *158*, 210-238.

Zhou, H., Qi, J., Yang, L., Shi, J., & Mo, P. (2022). Joint optimization of train scheduling and rolling stock circulation planning with passenger flow control on tidal overcrowded metro lines. *Transportation Research Part C: Emerging Technologies*, *140*, 103708.

Zhu, Y., Wang, H., & Goverde, R. M. (2020, September). Reinforcement Learning in Railway Timetable Rescheduling. In *2020 IEEE 23rd International Conference on Intelligent Transportation* Systems *(ITSC)* (pp. 1-6). IEEE.

APPENDICES

A.1 Classification of articles on train scheduling problem

No Author	Problem type	Infrastructure	Objective(s)	Model Structure	Solution Aproach(es)
1 Yang, L., Gao, Z., & Li, K. (2010).	Train scheduling (timetabling)	Network, double, single-tracked line	Minimizing total passengers’ trip time	MIP	Branch-and-bound algorithm
2 Burdett, R. L., & Kozan, E. (2010)	Train scheduling (timetabling)	Network, Single-tracked line	Minimizing makespan	Disjunctive graph model	Hybrid job shop approach
3 Meng, X., Jia, L., & Qin, Y. (2010)	Train rescheduling (dispatching)	Double-tracked line	Minimizing total delay time, optimizing stability	MIP in MATLAB	Improved particle swarm algorithm
4 Corman, F., D’Ariano, A., Pacciarelli, D., & Pranzo, M. (2010)	Train rescheduling (dispatching)	No clear information	Minimizing delay	Mixed integer disjunctive	Tabu search
5 Narayanaswami, S., & Rangaraj, N. (2011)	Review on scheduling and rescheduling (Review)				
6 Canca, D., Zarzo, A., Algaba, E., & Barrena, E. (2011)	Train scheduling (timetabling)	Network	Minimizing passenger waiting time and number of trains	Mixed integer non-linear programming	ε-constrains method
7 Liu, S. Q., & Kozan, E. (2011)	Train scheduling (timetabling)	Single-tracked line	Minimizing makespan	Mathematical model	Best-Insertion-Heuristic algorithm, Generic constructive algorithm
8 Mu, S., & Dessouky, M. (2011)	Train scheduling (timetabling)	Network	Minimizing total train delay	Mathematical model	LtdFlePath and GA + FixedPath algorithms, Decomp and Parallel algorithm
9 Acuna-Agost, R., Michelon, P., Feillet, D., & Gueye, S. (2011)	Railway rescheduling problem (dispathcing)	Network	Minimizing total rescheduling cost (delays,changing tracks,unplaned stops)	MIP in C# with CPLEX	Local-branching-type cuts, Right-shift rescheduling, Local search

10 Acuna-Agost, R., Michelon, P., Feillet, D., & Gueye, S. (2011)	Train rescheduling (dispatching)	Network- double-tracked line, double-tracked station	Minimizing total rescheduling cost (delays,changing tracks,unplaned stops)	MIP	SAPI (Statistical Analysis of Propagation of Incidents), heuristic approach
11 Cacchiani, V., & Toth, P. (2012)	Time tabling review (Review)				
12 Yang, X., Li, X., **Gao**, Z., Wang, H., & Tang, T. (2012)	Train scheduling (timetabling)	Network	Maximizing overlapping	Mathematical model in MATLAB	Genetic algorithm, graph theory
13 Özgür Yalçınkaya, G. Mirac Bayhan (2012)	Train scheduling (timetabling)	Network	To obtain a feasible timetable for trains	Stochastic simulation model in ARENA	Simulation
14 Yuri N. Sotskov, Omid Gholami (2012)	Train scheduling (timetabling)	Network, single-tracked	Minimizing total weighted tardiness of trains	Disjunctive graph model	Shifting bottleneck algorithm, heuristic, Job-shop scheduling
15 Keisuke Sato, Kei Tamura, Norio Tomii (2013)	Timetable rescheduling (dispatching)	Network	Minimizing convenience to customer	MIP in GUROBI	Shortest path algorithm, other counter measures
16 Almodóvar, M., García-Ródenas (2013)	Train rescheduling (timetabling)	Network, single-tracked line	Minimizing total time in system for passengers	Discrete-event simulation model in MATLAB	Greedy heuristic, on-line optimization
17 Hu, H., Li, K., & Xu, X. (2013)	Train scheduling (timetabling)	Network	Minimizing the energy and emission cost, minimizing total passenger-time	Mathematical model in GAMS, LINGO	Fuzzy multi-objective optimization algorithm
18 Cacchiani, V., Huisman, D., Kidd, M., Kroon, L., Toth, P., Veelenturf, L., & Wagenaar, J. (2014)	Review on train rescheduling (dispatching)				
19 Espinosa-Aranda, J. L., García-Ródenas, R., Cadarso, L., & Marín, Á. (2014)	Train scheduling (timetabling)	Network	Maximizing profit	MIP in CPLEX	Linearisation

20 Li, X., & Lo, H. K. (2014)	Train scheduling (timetabling)	Network	Minimizing energy consumption	Mathematical model	Dynamic train scheduling and control framework, linear approximation method
21 Dollevoet, T., Corman, F., D'Ariano, A., & Huisman, D. (2014)	Train scheduling (timetabling) and delay management	Network	Minimizing passenger delays	MIP	Iterative optimization framework
22 Xiang Li, Biying Shou, Dan Ralescu (2014)	Train rescheduling (dispatching)	Network, double tracked line	Minimizing delay and track changing cost	MIP in LINGO and MATLAB	Greedy algorithm
23 Lingyun Meng, Xuesong Zhou (2014)	Train rescheduling (dispatching)	N-tracked line	Minimizing delay cost	MIP in CPLEX and C++	Lagrangian relaxation solution framework with an efficient shortest path algorithm
24 Fang, W., Yang, S., & Yao, X. (2015)	Review on train rescheduling (Review)				
25 Yang, X., Li, X., Ning, B., & Tang, T. (2015)	Train scheduling (timetabling)	Network	Minimizing energy consumption and travel time	Integer programming model	Adaptive genetic algorithm, optimal train control algorithm
26 Qi, J., Yang, L., Gao, Y., Li, S., & Gao, Z. (2016)	Train scheduling (timetabling)	Single-tracked line	Minimizing total travel time	S-LMIP in GAMS	Local search based heuristics
27 Cacchiani, V., Furini, F., & Kidd, M. P. (2016)	Train scheduling (timetabling)	Network	To determine conflict-free timetables	ILP in C++	LP relaxation, heuristic, dynamic programming
28 Huang, Y., Yang, L., Tang, T., Cao, F., & Gao, Z. (2016)	Train scheduling (timetabling)	Network	Minimizing energy consumption and passenger travel time	Mathematical model in C#	Genetic algorithm
29 Yang, L., Qi, J., Li, S., & Gao, Y. (2016)	Train scheduling (timetabling)	Network	Minimizing the total delay and dwelling time	MILP in GAMS, CPLEX	Collaborative optimization

30 Zhan, S., Kroon, L. G., Zhao, J., & Peng, Q. (2016)	Train rescheduling (dispatching)	Network	Minimizing the total weighted sum of train service cancelations and train service deviations	MILP	Rolling horizon approach
31 Yin, J., Yang, L., Tang, T., Gao, Z., & Ran, B. (2017)	Train scheduling (timetabling)	Network	Minimizing energy consumption	MILP in CPLEX, C++	Lagrangian relaxation-based heuristic algorithm
32 Lamorgese, L., Mannino, C., & Natvig, E. (2017)	Train scheduling (timetabling)	Network	Minimizing the deviation, running time for trains, maximize the number of trains	MILP in CPLEX	Logic-based Benders′ decomposition
33 Corman, F., D’Ariano, A., Marra, A. D., Pacciarelli, D., & Samà, M. (2017)	Train scheduling (timetabling)	Network	Minimizing total time spent in the system by all passengers	MILP in CPLEX	Heuristic
34 Fischetti, M., & Monaci, M. (2017)	Train rescheduling (dispatching)	Network	Minimizing average delay	MILP in CPLEX	Heuristic, Randomized variable fixing
35 Shi, J., Yang, L., Yang, J., & Gao, Z. (2018)	Train scheduling (timetabling)	Network	Optimization for train timetable and passenger flow control	ILP in CPLEX	Heuristic
36 Liu, R., Li, S., Yang, L., & Yin, J. (2018)	Train scheduling (timetabling)	Network	Minimizing energy consumption	Mathematical model	Approximate Dynamic Programming Approach
37 Jiang, Y., & Zhou, X. (2018)	Train rescheduling (dispatching)	Network	Minimizing the processing time and the train operation time, minimizing the difference between the original schedule and the rescheduled one	Mathematical model	Genetic algorithm

38 Shakibayifar, M., Sheikholeslami, A., & Jamili, A. (2018)	Train rescheduling (dispatching)	Network	Minimizing total average delay	Simulation-based optimization framework	Neighborhood search, disturbance management model
39 Shakibayifar, M., Sheikholeslami, A., & Corman, F. (2018)	Train rescheduling (dispatching)	Network	Minimizing the total train delay time	Simulation-based optimization model in	Neighborhood search
40 Gholami, O., & Törnquist Krasemann, J. (2018)	Train rescheduling (dispatching)	Network	Minimizing total train delay	Alternative graph	Heuristic
41 Wen, C., Huang, P., Li, Z., Lessan, J., Fu, L., Jiang, C., & Xu, X. (2019)	Review on train rescheduling (Review)				
42 Wang, Y., Wei, Y., Zhang, Q., Shi, H., & Shang, P. (2019)	Train scheduling (timetabling)	Network	Minimizing the number of failed transfer passengers in the last train service and the transfer waiting time for the first train service	MILP in CPLEX	Lagrangian relaxation–based decomposition, solution framework Lagrangian relaxation–based decomposition, solution framework
43 Hassannayebi, E., Boroun, M., Jordehi, S. A., & Kor, H. (2019)	Train scheduling (timetabling)	Network	Minimizing the passenger wait time	Hybrid discrete-event simulation and response surface methodology (RSM) in Enterprise Dynamics simulation software and MATLAB	Meta-model simulation-based optimization
44 Wang, Y., Wei, Y., Zhang, Q., Shi, H., & Shang, P. (2019)	Train scheduling (timetabling)	Network	Minimizing the number of failed transfer passengers in the last train service and the transfer waiting time for the first train service	MILP in CPLEX	Lagrangian relaxation–based decomposition, solution framework Lagrangian relaxation–based decomposition, solution framework

45 Gianmarco Garrisi, Cristina Cervelló-Pastor (2020)	Train scheduling (timetabling), train scheduling on railway	Single-tracked line, Double-tracked station (Multi-Platform station)	To create a model that optimizes scheduling	MILP	Heuristic, genetic algorithm
46 Cheng-shuo Ying, Andy H.F. Chow, Kwai-Sang Chin (2020)	Train scheduling (timetabling)	Single-tracked line	Minimizing passenger waiting time and train operating cost	Markov decision process (MPD)	Deep reinforcement learning.
47 Huimin Zhang, Shukai Li, Yihui Wang, Lixing Yang, Ziyou Gao (2020)	Train rescheduling and track emergency maintenance	Double-tracked line	Efficient operation and low maintenance cost	MINLP	Lagrangian relaxation, Rolling horizon
48 Cheng-shuo Ying, Andy H.F. Chow, Kwai-Sang Chin (2020)	Train Scheduling (timetabling)	Single-tracked line	Minimizing passenger waiting time and train operating cost	Markov decision process (MPD)	Deep reinforcement learning
49 Abels, D., Jordi, J., Ostrowski, M., Schaub, T., Toletti, A., & Wanko, P. (2021).	Train Scheduling (timetabling)	Single-tracked line	Minimizing total delay time	Answer Set Programming (ASP)	Answer Set Programming (ASP)
50 Gong, C., Shi, J., Wang, Y., Zhou, H., Yang, L., Chen, D., & Pan, H. (2021)	Scheduling (timetabling)	Single-tracked line	Minimizing total total service cost	Integer Linear Programming	Variable neighborhood search
51 Wang, Y., Zhao, K., D'Ariano, A., Niu, R., Li, S., & Luan, X. (2021).	Scheduling (timetabling)	Single-tracked line	Minimizing passenger waiting time and operating costs	Simulation model	ε-constrains method
52 Zhang, Q., Lusby, R. M., Shang, P., & Zhu, X. (2022).	Scheduling (timetabling)	Single-tracked line	Minimizing train delays and missed connections	Binary integer program	Heuristic

53 Zhou, H., Qi, J., Yang, L., Shi, J., & Mo, P. (2022)	Train Scheduling (timetabling)	Single-tracked line	Minimizing passenger waiting time and operating costs	MILP	Tabu search
54 Wang, X., Lv, Y., Sun, H., Xu, G., Qu, Y., & Wu, J. (2023)	Train Rescheduling (dispatching)	Single-tracked line	Minimizing passenger waiting time and operating costs	MILP	Heuristic
55 This thesis	Scheduling (timetabling)	Single-tracked line	Minimizing average train travel time	Simulation-based optimization	Simulation integrated hybrid metaheuristic

A.2 Result of SimNGA, SimGA,SimGAb and SimGAfs

			SimNGA			SimGA			SimGAb			SimGAfs		
PS	**CR**	**MR**	**Exprmnt. Number**	**Solution**	**Fitness Value**	**Exprmnt. Number**	**Solution**	**Fitness Value**	**Exprmnt. Number**	**Solution**	**Fitness Value**	**Exprmnt. Number**	**Solution**	**Fitness Value**
10	0.4	0.1	**1**	6,3,1,2,4,1,2,6,5	21727	**28**	6,3,4,2,4,4,4,1,4	21704	**55**	1,3,1,2,2,4,4,1,4	21701	**82**	1,6,4,2,4,1,5,3,5	21697
10	0.4	0.2	**2**	6,3,4,2,2,4,5,6,4	21693	**29**	1,3,1,2,4,4,5,6,4	21698	**56**	1,3,1,2,4,4,4,1,4	21696	**83**	1,3,1,2,4,4,4,1,4	21696
10	0.4	0.3	**3**	1,6,1,2,2,5,4,1,4	21702	**30**	1,3,1,2,4,4,3,6,4	21703	**57**	1,3,1,2,4,4,4,1,4	21696	**84**	1,3,1,2,4,4,4,1,4	21696
10	0.6	0.1	**4**	6,3,1,2,4,4,5,6,4	21708	**31**	6,3,4,2,2,4,4,1,4	21689	**58**	1,3,5,2,2,4,4,1,4	21690	**85**	1,3,1,2,2,4,4,1,4	21701
10	0.6	0.2	**5**	1,3,1,2,4,1,2,6,5	21718	**32**	1,3,5,2,4,4,5,1,4	21694	**59**	1,3,1,2,2,4,4,1,4	21701	**86**	1,3,1,2,2,4,4,1,4	21701
10	0.6	0.3	**6**	1,3,1,2,4,4,5,6,4	21698	**33**	6,3,4,2,2,5,5,1,4	21690	**60**	1,3,1,2,2,4,4,1,4	21701	**87**	1,3,5,2,2,4,4,1,4	21690
10	0.8	0.1	**7**	1,3,1,2,2,4,2,1,4	21706	**34**	6,3,5,2,4,4,5,3,4	21708	**61**	6,3,4,2,2,4,4,1,4	21689	**88**	6,3,4,2,2,4,4,1,4	21689
10	0.8	0.2	**8**	1,3,1,2,4,1,5,6,4	21714	**35**	1,6,5,2,4,4,5,1,4	21694	**62**	6,3,4,2,2,4,4,1,4	21689	**89**	6,3,1,2,2,4,4,1,4	21711
10	0.8	0.3	**9**	1,3,4,2,4,1,2,1,4	21705	**36**	6,3,4,2,2,4,4,1,4	21689	**63**	6,3,1,2,2,4,4,1,4	21711	**90**	6,3,1,2,2,4,4,1,4	21711
20	0.4	0.1	**10**	1,3,2,2,4,4,4,6,5	21706	**37**	6,3,1,2,2,4,2,6,4	21716	**64**	6,3,5,2,2,1,2,6,4	21704	**91**	6,6,5,2,2,1,2,6,4	21704
20	0.4	0.2	**11**	1,3,4,2,4,4,2,3,5	21708	**38**	3,6,5,2,5,4,4,1,4	21784	**65**	6,3,1,2,5,4,4,1,4	21786	**92**	6,3,5,2,2,1,2,1,4	21700
20	0.4	0.3	**12**	1,3,1,2,4,4,4,6,5	21698	**39**	6,3,1,2,5,1,4,6,4	21802	**66**	6,3,4,2,2,1,2,6,4	21704	**93**	6,3,5,2,2,1,4,1,4	21690
20	0.6	0.1	**13**	6,3,4,2,4,1,2,6,5	21718	**40**	6,3,1,2,4,4,5,1,4	21706	**67**	1,3,1,2,2,1,5,6,4	21720	**94**	1,3,1,2,2,1,5,6,4	21720
20	0.6	0.2	**14**	1,3,3,2,4,4,2,3,5	21710	**41**	6,3,5,2,5,1,2,1,4	21793	**68**	6,6,5,2,2,1,2,6,4	21704	**95**	6,3,5,2,2,1,2,6,4	21704
20	0.6	0.3	**15**	1,3,1,2,4,1,5,3,4	21714	**42**	1,3,1,2,4,4,5,1,5	21696	**69**	6,3,5,2,2,1,2,6,4	21704	**96**	6,3,5,2,2,5,2,6,4	21703
20	0.8	0.1	**16**	1,6,1,2,5,4,2,1,5	21781	**43**	6,3,5,2,2,1,2,1,4	21700	**70**	6,6,5,2,2,1,2,6,4	21704	**97**	6,3,5,2,2,1,2,6,4	21704
20	0.8	0.2	**17**	1,3,1,2,4,1,2,6,5	21718	**44**	3,3,5,2,4,1,5,6,4	21704	**71**	6,6,5,2,2,1,2,6,4	21704	**98**	6,6,5,2,2,1,5,6,4	21694
20	0.8	0.3	**18**	1,3,4,6,4,1,2,1,5	21720	**45**	1,3,1,2,4,5,4,6,4	21699	**72**	6,3,5,2,2,1,2,6,4	21704	**99**	6,6,5,2,2,1,2,6,4	21704
30	0.4	0.1	**19**	6,3,4,2,5,4,5,3,4	21786	**46**	6,3,4,2,5,1,2,1,4	21793	**73**	6,3,1,2,4,4,4,1,4	21706	**100**	6,3,1,2,4,4,4,1,4	21706
30	0.4	0.2	**20**	1,3,1,2,5,5,2,1,4	21781	**47**	6,3,4,2,4,5,2,6,4	21719	**74**	6,3,1,2,4,4,4,1,4	21706	**101**	3,6,5,2,2,1,2,6,5	21704
30	0.4	0.3	**21**	6,3,2,6,4,4,3,1,4	21735	**48**	6,3,1,2,4,1,2,1,4	21724	**74**	6,3,4,2,4,4,4,1,4	21704	**102**	6,3,4,2,4,4,2,1,4	21714
30	0.6	0.1	**22**	1,3,1,2,5,5,4,6,4	21777	**49**	6,3,4,2,4,1,2,1,4	21714	**76**	6,3,4,2,4,4,4,1,4	21704	**103**	6,3,4,2,4,4,4,1,4	21704
30	0.6	0.2	**23**	6,3,1,2,4,1,2,6,5	21727	**50**	6,3,5,2,2,4,2,1,4	21699	**77**	6,3,5,2,2,1,2,1,4	21700	**104**	6,3,4,2,4,4,4,1,4	21704
30	0.6	0.3	**24**	6,3,2,2,4,4,4,6,5	21724	**51**	6,3,5,2,4,1,2,3,4	21718	**78**	6,3,4,2,4,4,4,1,4	21704	**105**	6,3,4,2,4,4,4,1,4	21704
30	0.8	0.1	**25**	6,3,4,2,4,1,2,6,2	21725	**52**	6,3,5,2,4,4,2,3,4	21718	**79**	1,6,4,2,2,5,2,3,5	21694	**106**	6,6,5,2,2,1,4,1,4	21690
30	0.8	0.2	**26**	1,3,1,2,4,1,5,3,4	21714	**53**	6,3,1,2,5,4,2,3,4	21789	**80**	6,3,4,2,2,1,2,6,5	21704	**107**	6,3,4,2,4,4,2,6,5	21798
30	0.8	0.3	**27**	1,3,1,2,4,1,5,3,4	21714	**54**	6,3,4,2,4,1,2,3,4	21718	**81**	6,3,4,2,4,4,5,1,4	21704	**108**	6,3,4,2,5,4,4,1,4	21825

A.3 Results of SimSA

				***E_0* =0.01**		***E_0* =0.007**		***E_0* =0.005**	
	E	***α***	***N***	**Solution**	**Fitness**	**Solution**	**Fitness**	**Solution**	**Fitness**
1	100	0.95	500	5,3,5,3,5,4,5,3,5	21810	5,2,4,2,3,3,4,2,3	21815	5,3,5,3,5,4,5,3,5	21810
2	100	0.95	1000	5,6,1,2,4,3,5,1,4	21722	5,6,1,2,4,3,5,1,4	21722	6,3,1,2,5,5,4,6,4	21787
3	100	0.95	1500	5,3,5,3,5,4,5,3,5	21810	5,6,3,2,3,5,1,4,3	21852	5,5,6,1,2,3,6,2,4	21836
4	100	0.97	500	5,3,2,5,2,1,1,3,6	21753	5,3,5,3,5,4,5,3,5	21810	1,6,1,2,5,1,5,6,5	21792
5	100	0.97	1000	5,6,6,2,3,1,5,1,4	21818	5,3,6,4,6,6,4,2,6	21790	5,6,3,2,3,5,1,4,3	21852
6	100	0.97	1500	5,3,6,3,4,1,4,3,4	21711	5,3,2,5,2,1,1,3,6	21753	6,3,1,2,5,5,4,6,4	21787
7	100	0.99	500	5,5,2,3,5,6,5,5,2	21857	5,3,5,3,5,4,5,3,5	21810	1,6,1,2,5,1,5,6,5	21792
8	100	0.99	1000	5,5,3,6,1,6,3,6,5	21843	5,5,3,6,1,6,3,6,5	21843	6,3,1,2,5,5,5,6,4	21787
9	100	0.99	1500	5,2,4,2,3,3,4,2,3	21815	5,6,1,2,4,3,5,1,4	21722	5,3,6,4,6,6,4,2,6	21790
10	300	0.95	500	3,2,2,2,5,5,5,2,1	21819	5,3,5,3,5,4,5,3,5	21810	6,2,6,2,5,4,3,3,4	21803
11	300	0.95	1000	5,2,1,6,6,5,6,6,6	21828	5,3,6,3,4,1,4,3,4	21711	6,3,2,2,4,5,6,5,5	21723
12	300	0.95	1500	5,5,6,1,2,3,6,2,4	21836	5,5,3,6,1,6,3,6,5	21843	5,3,5,2,4,4,5,1,4	21704
13	300	0.97	500	5,2,2,6,5,6,3,5,5	21844	5,3,6,4,6,6,4,2,6	21790	5,6,5,2,5,5,5,3,5	21788
14	300	0.97	1000	1,6,4,2,5,5,2,6,5	21786	1,6,1,2,5,1,5,6,5	21792	5,5,3,6,1,6,3,6,5	21843
15	300	0.97	1500	3,6,1,2,5,4,4,1,4	21786	3,6,5,2,5,1,2,1,5	21793	5,3,6,3,4,1,4,3,4	21711
16	300	0.99	500	5,3,6,4,6,6,4,2,6	21790	3,6,4,2,5,4,2,1,4	21793	6,6,1,2,5,4,4,6,4	21786
17	300	0.99	1000	5,3,5,3,5,4,5,3,5	21810	5,6,5,2,5,5,5,3,5	21788	3,3,5,2,5,5,5,3,5	21786
18	300	0.99	1500	3,6,4,2,5,4,2,1,4	21793	5,3,5,3,5,4,5,3,5	21810	6,3,1,2,5,5,5,6,4	21787
19	500	0.95	500	6,3,2,2,4,5,6,5,5	21723	5,6,3,2,3,5,1,4,3	21852	3,3,5,2,5,5,4,3,5	21787
20	500	0.95	1000	6,3,1,2,5,5,4,6,4	21787	5,5,6,1,2,3,6,2,4	21836	6,3,1,2,5,5,2,6,4	21790
21	500	0.95	1500	1,6,1,2,5,1,5,1,5	21791	3,3,4,2,5,1,5,1,5	21784	5,5,3,6,1,6,3,6,5	21843
22	500	0.97	500	1,6,1,2,5,1,4,1,5	21791	5,3,6,4,6,6,4,2,6	21790	5,3,5,3,5,4,5,3,5	21810
23	500	0.97	1000	1,3,1,2,5,1,5,3,5	21792	5,3,5,3,5,4,5,3,5	21810	6,3,1,2,5,5,4,6,4	21787
24	500	0.97	1500	1,6,1,2,5,1,5,6,5	21792	5,6,3,2,3,5,1,4,3	21852	3,3,1,2,5,5,4,6,4	21787
25	500	0.99	500	1,6,1,2,5,1,4,6,5	21792	5,5,3,6,1,6,3,6,5	21843	6,3,2,2,4,5,6,5,5	21723
26	500	0.99	1000	3,6,4,2,5,5,2,6,5	21782	5,3,6,3,4,1,4,3,4	21711	3,3,4,2,5,1,5,1,5	21784
27	500	0.99	1500	1,6,4,2,5,4,2,1,4	21784	5,6,4,2,4,1,5,1,4	21704	5,2,4,2,3,3,4,2,3	21815
28	1000	0.95	500	3,3,4,2,5,4,5,1,5	21784	5,6,2,4,2,1,2,5,3	21776	5,3,6,3,4,1,4,3,4	21711
29	1000	0.95	1000	3,3,4,2,5,1,5,1,5	21784	5,6,6,2,3,1,5,1,4	21818	5,3,6,3,4,1,4,3,4	21711
30	1000	0.95	1500	3,3,4,2,5,5,5,1,5	21785	3,3,4,2,5,5,2,6,5	21782	5,6,3,2,3,5,1,4,3	21852
31	1000	0.97	500	1,3,1,2,5,1,5,3,5	21792	6,3,1,2,5,5,5,6,4	21787	3,6,4,2,5,4,2,1,4	21793
32	1000	0.97	1000	3,3,5,2,5,5,5,3,5	21786	1,6,4,2,5,5,2,6,5	21786	1,6,1,2,5,1,5,6,5	21792
33	1000	0.97	1500	1,6,4,2,5,5,2,6,5	21786	5,6,5,2,5,5,5,3,5	21788	5,6,5,2,5,5,5,3,5	21788
34	1000	0.99	500	1,3,1,2,5,1,5,3,5	21792	3,6,4,2,5,4,2,1,4	21793	5,2,1,6,6,5,6,6,6	21828
35	1000	0.99	1000	5,2,6,2,3,4,6,3,4	21847	5,2,4,2,3,3,4,2,3	21815	5,5,3,6,1,6,3,6,5	21843
36	1000	0.99	1500	6,5,1,6,4,4,5,4,4	21744	5,6,2,4,2,1,2,5,3	21776	5,3,6,3,4,1,4,3,4	21711

A.4 Results of SimHA

				maxNIter=100		maxNIter =150		maxNIter =200	
	PS	**CR**	**MR**	**Solution**	**Fitness Value**	**Solution**	**Fitness Value**	**Solution**	**Fitness Value**
1	10	0.4	0.1	6,3,4,2,4,4,5,3,5	21711	1,6,1,2,5,4,5,1,5	21726	1,3,1,2,4,4,5,3,5	21698
2	10	0.4	0.2	6,2,1,2,5,4,5,2,5	21703	1,3,1,2,4,4,2,1,5	21683	1,3,1,2,5,4,2,6,5	21780
3	10	0.4	0.3	1,3,1,2,4,4,5,1,4	21732	1,3,1,2,5,4,4,6,5	21708	1,3,1,2,4,4,5,6,4	21698
4	10	0.6	0.1	1,3,1,2,4,1,4,6,5	21714	1,3,1,2,4,4,5,6,5	21715	1,3,1,2,4,4,5,3,4	21715
5	10	0.6	0.2	1,3,1,2,4,4,4,1,4	21732	1,3,1,2,4,4,4,6,4	21715	1,3,1,2,4,1,4,6,4	21707
6	10	0.6	0.3	1,3,1,2,4,4,2,6,4	21665	1,3,1,2,4,4,5,6,5	21715	1,3,1,2,4,4,4,3,4	21715
7	10	0.8	0.1	1,3,1,2,4,4,5,1,4	21732	1,3,1,2,2,5,4,1,5	21679	1,3,1,2,4,4,5,6,2	21697
8	10	0.8	0.2	1,3,1,2,4,4,4,6,5	21715	1,3,1,2,4,4,4,6,4	21715	1,3,1,2,2,5,2,6,5	21688
9	10	0.8	0.3	1,3,1,2,2,5,2,3,5	21708	1,3,1,2,4,4,2,6,4	21665	1,3,1,2,2,5,2,3,5	21708
10	20	0.4	0.1	1,6,4,2,5,4,5,3,5	21701	1,3,1,2,4,4,5,6,4	21698	1,3,5,2,4,4,5,1,4	21694
11	20	0.4	0.2	1,3,1,2,4,4,4,6,4	21715	1,3,4,2,2,4,4,1,4	21679	1,3,1,2,4,4,4,6,4	21715
12	20	0.4	0.3	1,3,1,2,4,4,2,6,4	21665	1,3,1,2,4,1,2,6,5	21718	1,6,4,2,4,1,5,3,5	21697
13	20	0.6	0.1	1,3,4,2,2,4,4,1,4	21679	1,3,1,2,4,4,5,6,4	21698	1,3,1,2,4,4,5,1,4	21732
14	20	0.6	0.2	1,3,4,2,2,4,4,1,4	21679	1,3,5,2,4,4,5,1,4	21694	1,3,1,2,4,4,4,6,4	21715
15	20	0.6	0.3	1,3,1,2,4,4,4,6,4	21715	1,3,1,2,2,5,2,6,5	21688	1,3,4,2,2,4,4,1,4	21679
16	20	0.8	0.1	1,3,3,2,4,4,2,3,5	21710	1,3,1,2,4,1,2,6,5	21718	1,3,3,2,4,4,2,3,5	21710
17	20	0.8	0.2	1,3,5,2,4,4,5,1,4	21694	1,3,2,6,4,1,4,1,4	21724	1,3,4,2,2,4,4,1,4	21679
18	20	0.8	0.3	1,3,1,2,4,4,5,6,4	21698	1,3,4,2,2,4,4,1,4	21679	6,3,4,2,5,1,2,1,4	21793
19	30	0.4	0.1	1,3,2,6,4,1,4,1,4	21724	1,3,1,2,5,5,2,1,4	21781	1,3,3,2,4,4,2,3,5	21710
20	30	0.4	0.2	1,3,1,2,4,4,5,1,4	21732	3,6,4,2,2,4,4,6,5	21693	1,3,2,6,4,1,4,1,4	21724
21	30	0.4	0.3	1,6,4,2,4,1,5,3,5	21697	1,3,5,2,4,4,5,1,4	21694	1,3,1,2,4,4,4,1,4	21732
22	30	0.6	0.1	6,3,1,2,5,4,4,1,4	21786	1,3,1,2,4,4,4,6,4	21715	1,3,1,2,5,5,2,1,4	21781
23	30	0.6	0.2	1,3,2,6,4,1,4,1,4	21724	1,6,4,2,4,1,5,3,5	21697	1,3,4,2,4,1,2,1,4	21705
24	30	0.6	0.3	1,6,4,2,2,5,4,1,5	21680	1,3,2,6,4,1,4,1,4	21724	1,6,4,2,4,1,5,3,5	21697
25	30	0.8	0.1	1,3,1,2,5,5,2,1,4	21781	6,3,4,2,5,1,2,1,4	21793	6,3,4,2,5,1,2,1,4	21793
26	30	0.8	0.2	1,3,2,6,4,1,4,1,4	21724	6,3,1,2,5,4,4,1,4	21786	1,3,1,2,5,5,2,1,4	21781
27	30	0.8	0.3	1,6,1,2,4,4,2,1,5	21702	1,3,1,2,4,4,4,1,4	21732	6,3,1,2,5,4,4,1,4	21786