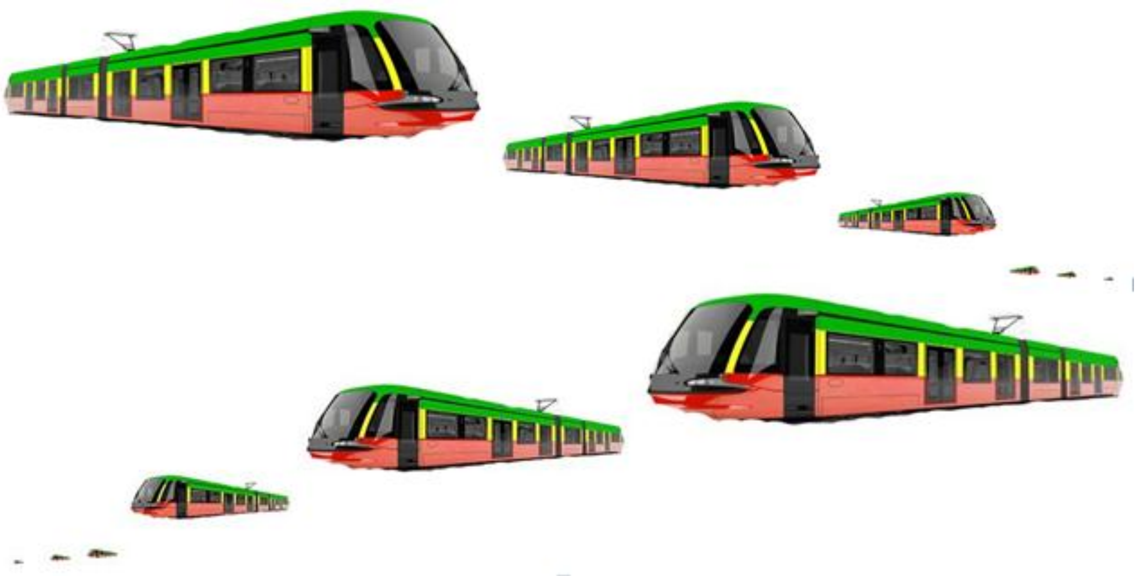
ADDIS ABABA UNIVERSITY

ADDIS ABABA INSTITUTE OF TECHNOLOGY

SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

COMPUTER BASED TRAIN SCHEDULE OPTIMIZER FOR AA-LRT SYSTEM

By Tesfaye Belachew Abebe



April, 2015
ADDIS ABABA, ETHIOPIA

ADDIS ABABA UNIVERSITY

ADDIS ABABA INSTITUTE OF TECHNOLGY

SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

ELECTRICAL RAILWAY ENGINEERING POST GRADUATE CLASS

Computer Based Train Schedule Optimizer for Addis Ababa LRT System

By Tefsaye Belachew Abebe

A thesis submitted to the School of Electrical and Computer Engineering in partial fulfilment of the requirement for the Degree of Master of Science in Railway Electrical Engineering

April, 2015

Addis Ababa, Ethiopia

ADDIS ABABA UNIVERSITY

ADDIS ABABA INSTITUTE OF TECHNOLGY

SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

Computer Based Train Schedule Optimize for Addis Ababa LRT System

By Tefsaye Belachew Abebe

Guided by: Dr. Yalemzewd Negash

ADDIS ABABA UNIVERSITY

ADDIS ABABA INSTITUTE OF TECHNOLGY

SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

Computer Based Train Schedule Optimize for Addis Ababa LRT System

By Tefsaye Belachew Abebe

APPROVED BY BOARD OF EXAMINERS

Dr. Yalemzewd Negash		
Dean, School of Electrical and Computer Engineering	Signature	Date
Dr. Yalemzewd Negash		
Advisor	Signature	Date
Dr. Yihenew Wondie		
External Examiner	Signature	Date
Abi Abate		
Internal Examiner	Signature	Date

ABSTRACT

This research paper work focuses on passenger train schedule modeling; scheduling algorithm which is used to solve the modeled problem and on the deployment of computer Based Train Schedule Optimizer (CBTSO) for Addis Ababa Light Rail Transit (LRT) system.

In railway system the guided movement of trains with highly sensitive safety issues together with the expensive nature of trains, railway infrastructures and train crews make the requirement of train and crew schedules inevitable. Schedules are the way to efficiently utilize the available railway resources and meet different constraints like traffic and operator's requirement constraints. There are two methods of scheduling namely manual scheduling (the traditional method) and computer based scheduling.

For more than 150 years, traditional scheduling was the typical scheduling method used in almost all railways found in the world. Only the occurrence of high performance computers in the 1990s and the advancements in the operation research methods and artificial intelligence methods offered the opportunity to introduce computer based scheduling methods.

Manual based train schedule requires skilled and experienced rail network planners. Further it requires a massive mental effort and takes long time to solve the problem. Further there is a rare chance to find such human powers in a newly starting railway companies like Ethiopian Railway Corporation (ERC). Considering these into account this research work concentrates to develop a Computer Based Train Schedule Optimizer (CBTSO) for Addis Ababa LRT which is currently in its final project stage.

In this paper a mathematical model used to formulate train schedule problem and the corresponding algorithm suitable to solve the schedule problem for Addis Ababa LRT system are presented. Eventually, a computer program that is planned to implement the algorithm will be discussed.

Key words: CBTSO, LRT, modeling, algorithm, Schedule, Optimization.

Declaration

I, the undersigned, declare that this thesis work, to the best of my knowledge and belief, is my original work, and all sources of materials used for the thesis work have been fully acknowledged.

Tesfaye Belachew Abebe		03, April 2015
Student Name	Signature	Submission Date

This Thesis has been submitted for examination with my approval as a university advisor.

Dr. Yalemzewd Negash		
Advisor Name	Signature	Date

Addis Ababa, Ethiopia

ACKNOWLEDGMENT

I would like to express my sincere appreciation to all people who contributed for this master thesis work in one way or another. Especially, I want to thank my advisor Dr. Yalemzewd Negash, Dean of School of Electrical and Computer Engineering department at AAiT and a coordinator of electrical railway engineering stream for Ethiopian Railway Institute Technology, for giving me the chance to work on this topic and his constant motivation, guidance, support, supervision and constructive suggestion throughout this research work.

I owe my gratitude to my best friends called Yared Taye, Asnake Tilahun and Yirga Badima for providing me valuable comments and suggestions to keep the research on track. My gratitude also goes to my family specially my younger sister who constantly encouraged and motivated me at the time of difficulties I encountered during the process of the research work. I wouldn’t wind up my acknowledgement before expressing my appreciation for Viviana, a colleague in my former employer. She was special person I worked with in my life who taught me how to handle different kinds of challenges in parallel.

I would also like to thank Ethiopian Railway Corporation for the sponsorship they offered me to study master’s degree in railway electrical engineering and employees working at Addis Ababa LRT system for providing me useful data for the research.
Eventually, but not the least, my acknowledgement go to my God for supporting me all along the way.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

IDE	Interactive software Development Environment
LC:	(Road Rail) Level Crossings
LRT	Light Rail Transit System
MINLP	Mixed Integer Nonlinear Programming
NP	Non Polynomial problems
N-S	North -South Line
NS27	Terminal station on North South corridor
PHA:	Preliminary Hazard Analysis [10]
PSR	Permanent Speed Restriction
RDMS	Relational Database Management System
RRR:	Rapid Risk Ranking
SATS	Scheduling Additional Train Services
SQL	Structural Query Language
TC	Topological Constraints
TPP	Train Platforming problem
TRAP	Train Route Allocation Problem
TRC	Train Route Constraints
TTP	Train Timetabling Problem
TTT	Train Timetable
VB.NET	Dot NET version of Visual Basic Programming language

CHAPTER 1

1. INTRODUCTION

Railway transportation has played a major role in the economic development of the world in last two centuries. It represented a major improvement in land transport technology and has obviously introduced important changes in the movement of freight as well as passenger transportations. Though there is no active railway transportation service currently in Ethiopia, over the last few years in most part of the world, railway traffic has increased considerably, which has created the need to optimize the use of railway infrastructures and to maintain robustness in railway scheduling. Thanks to developments in computer science and advances in the fields of optimization and intelligent resource management, railway managers can optimize the use of available infrastructures and railway capacity. The term "robust" here refers to the ability to resist to "schedule imprecision" during interruptions. In this paper Computer Based Train Schedule Optimizer (CBTSO) for Addis Ababa LRT system will be presented. In the following sections a background for train scheduling is described in detail.

Train scheduling problem is basically an optimization problem which is computationally difficult to solve. Several models and methods have been analyzed to solve it **[1]**, **[2]**, **[6]**, **[8]**, **[9]**, **[13]**,**[15]**, **[22]**, **[28]**,**[29]**, **[30]**,**[31]**, **[33]**, **[37]**,**[41]**, **[42]**,**[44]** and **[46]**. The main purpose of this paper is to develop a model and design the corresponding suitable algorithm to solve train scheduling problem for Addis Ababa Light Rail Transit System (LRT) **[7]** which is managed by Ethiopian Railway Corporation (ERC). Further the algorithm is implemented using a computer software application. This application is coded and deployed using an integrated development environment called Microsoft Visual Studio 2010 **[20]**. The programming language to be used for this purpose are Visual C Sharp dot Net 4.0 (C#.NET) programming language **[21]** for the interface part and Microsoft SQL Server 2008 for the database part **[38]**. Solution for the train scheduling optimization problem will be solved by integrating optimization tool called IBM CPLEX concert technology for .NET application **[17]**, **[18]** with the visual studio. The following subsections of this document describes about the current condition of the targeted LRT system and about Scheduling in general.

1.1 BACKGROUND

1.1.1. RAILWAY AND ROAD TRANSPORT IN ADDIS ABABA CITY

Currently there is no railway transportation service in Ethiopia. However; according to Ethiopian Railway Corporation report, the LRT project is in its finishing work when this research work is completed and it is expected to begin operation before the end of 2007 E.C. In Addis Ababa, capital city of Ethiopia there is a fast population growth and small road network coverage. At the time of the City's 125th year of founding anniversary in 2013 Addis Ababa City Road Authority (AACRA) reported, in their official website, that the city's road coverage was not more than 13.7%. By 2020 this coverage is expected to be 20%. The completion of the LRT project on the other hand is believed to complement the existing road transport service and reduce the existing transportation problem in the city. The rough topology of Addis Ababa LRT is illustrated on figure 1. For more detailed topology for both corridors refer Appendices A and B.

As you can see from the LRT topology the project is divided into two phases. In the first phase, part of the route in East-West colored in blue and part of the route in the South-North colored in red will be completed whereas the remaining drawn in a broken line will be constructed in the second phase of the project. You can find the softcopy of with the right colors from Addis Ababa Institute of Technology. After completion of the first phase of the project Addis Ababa rail network coverage will be 34.25 km in two corridors that divides the city into four parts. The two corridors are called North-South line and East-West line. Their total lengths are 16.9 km and 17.35 km respectively and these corridors share a common track of about 2.7km which starts from Addis Ababa Stadium and ends at Lideta station as show in the rough topology figure 1.1.

From Addis Ababa LRT daily passenger flow forecast report **[7]** on the East-West line, you can also see on the table below, there are around 11,202 passengers in the initial stage, 23,435 passengers in the short term and 38,460 passengers in the long term during the non-peak hours and 86,171 passengers in the initial stage, 180,270 in the short term and 295,847 passengers in the long term plans during the peak hours i.e. 7:00AM to 9:00AM and from 5:00PM to 7:00PM. If this is approximated for passenger flows per year it will be 4.1, 8.6 and 14.0 million passengers in the initial stage, short term and long term respectively during the non-peak hours. Similarly for peak hours the annual passenger flow is expected to be 31.5, 65.8 and 108.0 million passengers respectively for the plan horizons stated above. According to ERC the capacity of the LRT is 80,000 Passengers/hour. These figures look very large but they are small compared to the number of passengers' demand in Addis Ababa. This implies that the current railway network needs to be expanded to satisfy passengers demand.

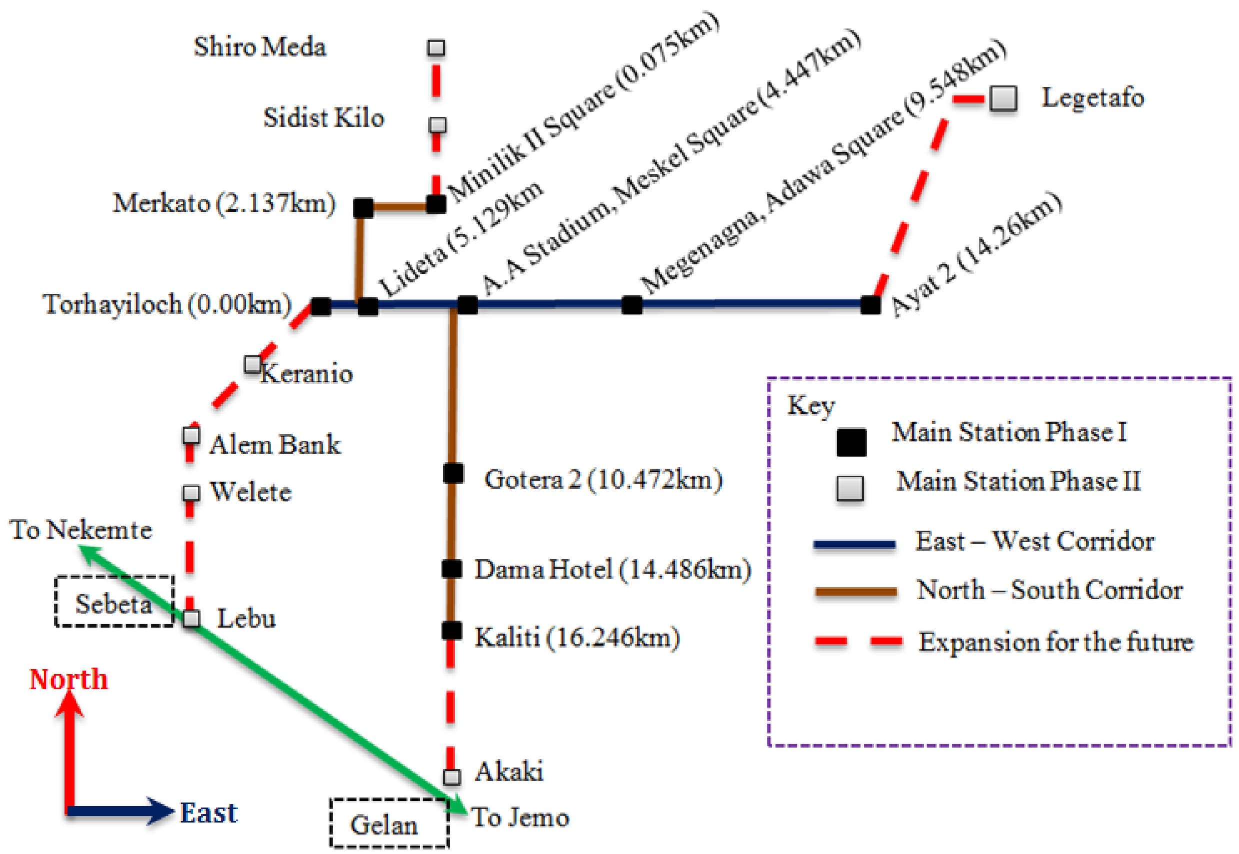


Figure 1.1: Addis Ababa LRT rough topology including future expansions

Note: The above diagram contains only (intermediate stations are not shown in this topology)

From trends of rail network operation the probability of rail network expansion work is very small. This is due to the requirement of huge amount of investment. Hence, in order to utilize the available infrastructure efficiently, on the other hand to satisfy passengers demand, the necessity of an optimized train schedule is inevitable. In railways operation train scheduling is one of the challenging planning process. It demands skilled rail network planners. Further the process of planning if it is needed to be constructed using manual method it takes from few weeks to few months period of time. As a result, it is cumbersome task to perform that way. So the significance of computer Based Train Schedule Optimizer (CBTSO) has no question in order to solve the varying requirement multi constraint problem within a reasonable computation time and generate many alternatives optimal or near optimal solutions. This paper presents the implementation of train's timetable problem solution for Addis Ababa LRT System.

Section/Route(From ~ To)	E-W Line Unidirectional Passenger flow per day forecast Unit: persons per day			E-W Line Unidirectional Passenger Flow Forecast during Peak Hours Unit: Persons/hr		
	Initial Stage	Short term	Long Term	Initial Stage	Short term	Long Term
Sebeta ~ Jimma JCT	~	6069	9962	~	789	1295
Jimma JCT ~ Torhailoch	~	11631	19092	~	1512	2482
Torhailoch ~ Lideta Light	15885	29031	47646	2065	3774	6194
Lideta Light ~ La Gare	38462	70308	115385	5000	9140	15000
La Gare ~ Adwa Square	19262	35208	57777	2504	4577	7511
Adwa Square ~ CMC	7031	12846	21085	914	1670	2741
CMC ~ Ayat	5531	10115	16600	719	1315	2158
Ayat ~ Legetafo	~	5062	8300	~	658	1079
Total	**86171**	**180270**	**295847**	**11202**	**23435**	**38460**

Table 1.1 Passenger flow forecast in the E-W line

The above figure shows passenger flow forecast in the East-West corridor only. See the Table 3 for the North-South line passenger flow forecast. It is difficult to manage such large throughput daily demand. Hence, to transport such large number of passengers daily is not an easy task under many constraints related to available resource usage. To handle these difficult tasks application of computer aided scheduling tool is preferable which simplifies the manager's daily activity in allocating trains in the different routes and fulfill the daily transportation demands.

Trains Operational Indices in the East- West Route			
Indices/Operation Stage		Initial Stage	Long Term
Operation Length (km)		17.1	37.7
Tramcar Marshalling		one unit	two unit
Rated Passengers (Persons/tramcar)		286	572
Operated Trains in peak hours (couple/hr)	Shared rail Section	20	30
	Other Sections	10	15
Minimum Traveling Interval (minutes)	Shared rail Section	3	2
	Other Sections	6	4
Transport Capacity (thousand passengers/hr)	Other Sections	5.7	17.1
	Operated Tramcars	19	59
Train Allocation(train)	Spare Tramcars & Maintenance Tramcars	2	6
	Total	**21**	**65**

Table 1.2: Route plan for East-West line

ERC's plan shows that there are eight railway route projects at a national level with a total length of approximately 5,060km. These routes will be constructed into two phases. In this paper only Addis Ababa LRT train scheduling is considered which is approximately 34.0km long. The following table illustrates the route plan of the LRT in the East-West corridor. This route plan designed by China

Railway Group Limited [7] will be used as a raw input for this research including the remaining corridor called North-South line, see Table 4.

N-S Line Unidirectional Passenger flow per day forecast Unit: persons/day				N-S Line Unidirectional Passenger Flow Forecast during Peak Hours Unit: Persons/hr		
Section/Route(From ~ To)	**Initial Stage**	**Short term**	**Long Term**	**Initial Stage**	**Short term**	**Long Term**
Shiromeda ~ Menelik II Square	~	12138	19923	~	1578	2590
Menelik II Square ~ Merkato	11623	21246	34862	1511	2762	4532
Merkato ~ Lideta Light	13008	23777	39015	1691	3091	5072
Lideta Light ~ La Gare	38462	70308	115385	5000	9140	15000
La Gare ~ Gotera	17708	32369	53123	2302	4208	6906
Gotera ~ Kaliti	16323	29846	48977	2122	3880	6367
Kaliti ~ Akaki		12646	20754	~	1644	2698
Akaki ~ Akaki Campus	~	5562	9131	~	723	1187
Total	**97124**	**207892**	**341170**	**11115**	**27026**	**44352**

Table 1.3: Passenger flow forecast for the North South Route

Trains Operational Indices in the North- South Route			
Indices/Operation Stage		Initial Stage	Long Term
Operation Length (km)		16.8	36
Tramcar Marshalling		one unit	two unit
Rated Passengers (Persons/tramcar)		286	572
Operated Trains in peak hours (couple/hr)	Section with Shared rail	20	30
	Other Sections	10	15
Minimum Traveling Interval (minutes)	Section with shared rail	3	2
	Other Sections	6	4
Transport Capacity(thousand passengers/hr)	Sections with shared rail	5.7	17.1
	Other Sections	2.9	8.6
Train Allocation(train)	Operated Tramcars	18	56
	Spare Tramcars & Maintenance Tramcars	2	6
	Total	20	62

Table 1.4: Route plan for North-South Line

Figure 1.2. Photographic view of interlocking area located at stadium

Note: It is the meeting point of NS and EW Lines

1.1.2. OVERVIEW OF SCHEDULING

In a general context [3] defines scheduling as a form of decision-making process that plays an important role in many disciplines. It is concerned with the allocation of scarce resources to activities with the objective of optimizing one or more performance measures. Depending on the situation, resources and activities can take on many different forms. Resources may be nurses in a hospital, bus drivers, machines in an assembly plant, computer CPUs, mechanics in an automobile repair shop, etc. In our case the term resource refers to trains, tracks, track sidings, stations, depots and train crews. Activities may be operations in a manufacturing process, duties of nurses in a hospital, executions of computer programs, car repairs in an automobile repair shop, and so on. There are also many different performance measures to optimize. One objective for instance may be the minimization of the mean flow time, while another objective may be the minimization of the number of jobs completed after their due dates.

Scheduling has been studied intensively for more than 50 years, by researchers in management, industrial engineering, operations research, and computer science **[3][14]**. There is now an astounding body of knowledge in this field though its way of implementation to the real world is found to be still challenging one particularly for large scale sized problems like train and train crew scheduling problems.

In railway companies scheduling is extensively applicable and it is the active area of research in railway sector. Train and train crew scheduling are a particular examples which are part of railway planning process. Train scheduling involves optimally assigning a set of locomotives to each train and finding the arrival and departure times at each station where trains are required to sop to load or unload passengers. Locomotive assignment and train's arrival & departure times are calculated so that the solutions satisfy varieties of business constraints, traffic constraints and user requirement constraints.

Everyday railway managers need to allocate a set of trains to different railway routes according to demand from their customers. If this task is carried out using manual method it will be difficult for railway system with large and complex networks. So nowadays the uses of computer aided scheduling tools are preferred. Hence, these software tools create blueprint for use by managers to assist them in making tactical, day-to-day decisions regarding rolling stock assignments.

In this research work a computer based scheduling tool, called CBTSO, is developed which can assist Addis Ababa LRT managers, despatchers, conductors and crews to manage their daily operations. Nowadays planning problems in railway systems become more manageable due to the algorithms better implementations on faster computers **[19]**. Especially solving huge linear programs, which is a substantial part of solving mixed integer problems, became much more efficient in the last 10 years **[8]**. Nevertheless a lot of mathematical work has to be done to solve the real world problem instances of a complex problem. Nearly every urban public transport (trams, bus) and a growing number of railway companies in the world use periodic timetable. The opposite one which is the non-periodic timetable is used for freight transport. The reason is that the second one usually does not involve a frequent train movement when compared to the passenger transport. As a result; implementing period timetable results movements of empty trains for freight transport. Whereas in the case of passenger transport inside urban involves frequent movement, hence employing periodic timetable is preferable. Some of the reasons are they are easy to be remembered by the passengers and easy to manage for the railway owners. Further they are also relatively easier for computation than the non-periodic once.

1.1.3. RAILWAY TRANSPORT PLANNING STAGES

In railway system there are several issues that require planning as discussed in [28]. Among them timetabling (also called railway scheduling) is the heart of all other plans. Hence other plans can be influenced by the timetable or vice versa. This section briefly describes the major railway planning problems, so as to clarify their relation with timetabling. The planning processes for a railway company, and the time dependencies between them. The following figure contains a mix of strategic planning problems, such as demand estimation and line planning, and more operational planning problems, such as rolling stock scheduling and crew scheduling. For the latter two, it is essential to also construct a strategic capacity plan, since acquiring new rolling stock and hiring new crews usually takes quite some time.

Planning a railway operation is generally complex task and hence it is partitioned into several problems that are solved sequentially. These problems and the sequence, in which they are solved, can be seen on Figure 1.3.

The train operating companies' planning process starts with the line planning based on the demand required to satisfy followed by the timetable design, rolling stock and crew scheduling and train platforming.

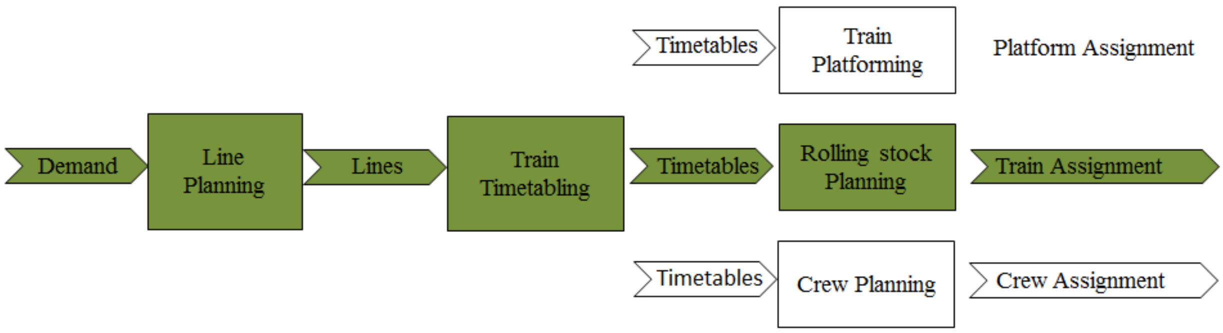


Figure 1.3: Planning overview of railway operartion

<u>*Demand Estimation:*</u> It is the first planning stage in railway planning process. The estimation of the demand for railway services lies at the basis of a railway system. Travel demand is estimated as the number of people that wish to travel from an origin to a destination. By estimating the travel demand for each possible Origin-Destination combination, a so-called OD-matrix of total travel demand is obtained. Passenger counts, passenger interviews, and ticket sales form the basic information for constructing an OD-matrix. Since Addis Ababa LRT service is not yet started there is no actual passenger travel demand. However, for this research work passenger demand forecasted by ERC in collaboration with CREC [7] will be used as demand estimation to calculate the dwell time at each station and to allocate trains along the two corridors on each direction.

Line Planning: After passenger demand has been estimated, one proceeds with deciding which train connections will be part of the railway system. The set of train connections which is operated is called the train line system. A train line is a direct train connection between an origin station and a destination station, via a certain route through the railway network. In this Planning Problem, the main input is OD- Matrix and the global railway infrastructure (network). Based on this input, selection of the most suitable potential train lines, connecting these origins and destinations, is undertaken. Apart from that, for each train line, a frequencies and capacities are specified, and a type, which determines the stations that the line calls at. In AA LRT, there are two lines or routes planned to be operated: the North-South line which connects the Kality depot with Piasa and the East-West line which connects Hayat depot with Torhayloch. Each corridor has 22 stations including the shared 5 stations between stadium and Lideta. The rough signal system rough schematic diagram of Addis Ababa LRT system can be seen from figures (Fig 4 and Fig 5).

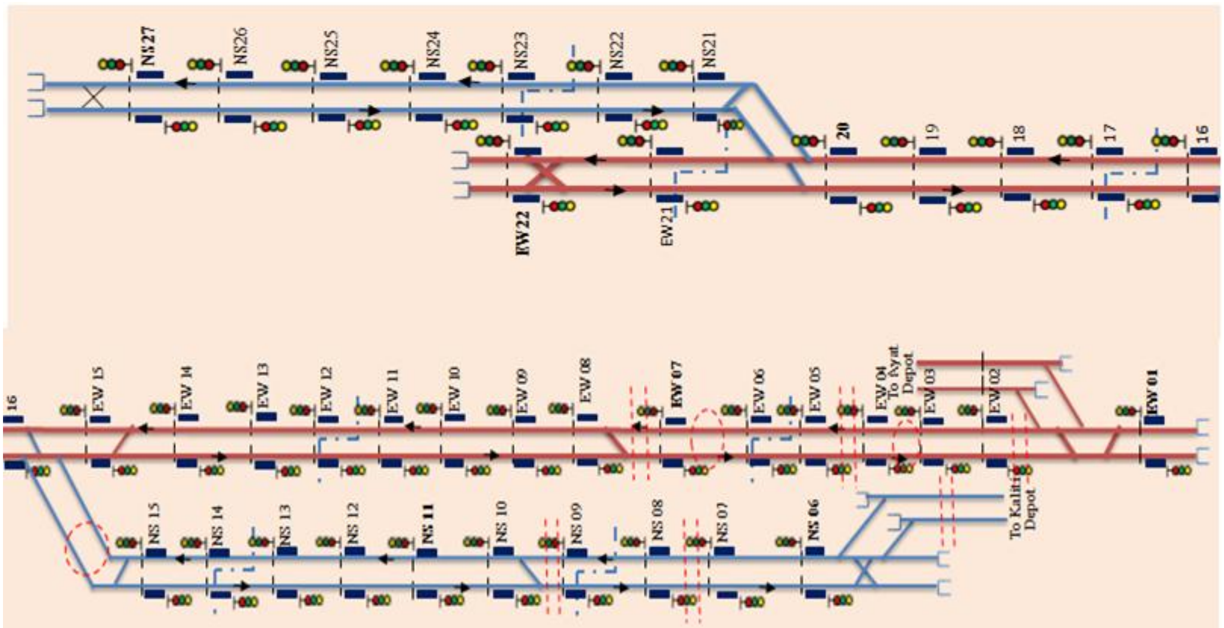


Figure 1.4: Rough schematic diagram of AA LRT main lines with all stations labeled

The line planning problem considers how to cover the railway network with lines, such that all traffic demand can be met, while meeting certain objectives. In this research work the objective is to minimize the total traveling time of passengers.

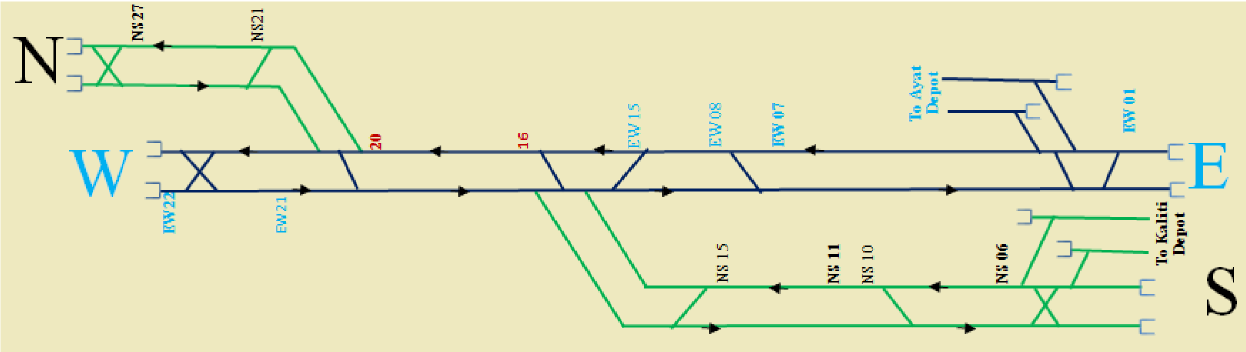


Figure 1.5: Rough schematic diagram of AA LRT main lines with only main stations labeled

Timetabling: Once the line plan is completed, a timetable for its train lines can be constructed. For a detailed description of the timetable planning process we refer to the previous section.

Rolling Stock Planning: The next planning problem to be solved is the assignment of train units to the train lines in the timetable. A railway company may own a variety of rolling stock types, for example, single deck and double deck units, wagons that need a locomotive, units that have their own engine, etc. When each train line has been assigned one or more types of rolling stock, a plan is constructed that specifies how many units each train consists of. Moreover, a unit does not have to be assigned to the same train line for the entire day. A unit may be used for one train line first, then for another, etc. During off-peak hours, not all rolling stock is in use, and the idle train units need to be shunted from the platforms to shunting yards in this case at depots. A shunting plan is constructed in a later planning phase. Also, each train unit needs to be taken out of circulation after having traveled a certain distance, in order to be maintained. A final planning phase considers how to adjust the flow of train units such that each unit is routed to a workshop when it requires maintenance. For AA LRT system a uniform train units which are composed of three cars are planned to be operated in the initial stage and two-units of this composition of trains are planned to be operated in the long term [7].

Crew Planning: Each train has to be manned by a driver and one or more conductors. This poses complex planning problems, since the crew plan needs to respect several complicated labor rules. The drivers and conductors have to return to their home bases by the end of the working day. Working shifts have to contain a meal break of at least half an hour, and may not contain a continuous period of work of more than five hours. More complicated rules also exist. A shift must contain some variation, so a driver or conductor may not be assigned to the same train line all day, going back-and-forth on that line. Drivers and conductors are allowed to transfer from one train to another only when sufficient buffer time for the transfer is available. Otherwise, a delayed arrival of the crew may result in a delayed departure of their next train. Furthermore, the crew schedule should incorporate various crew member characteristics, such as rostering qualifications, individual requests of crew members, and the past rosters of crew members. These past rosters are of importance for the labor rules. Finally, taking all these rules and characteristics into account, a railway operator aims at constructing a crew plan that meets certain objectives, such as minimizing the number of duties that cover all the work, or maximizing crew satisfaction.

The flow of the overall planning process as shown in Fig 3 above arises because certain processes provide the input for others. However, the planning flow is not as linear as it may seem, since feedback loops between the several planning stages exist. For example, the timetable may be changed to improve the rolling stock circulation or the crew schedules.

These dependencies between the railway planning processes show the importance of timetabling for the planning chain as a whole. If fast methods exist to construct good timetables, one has to take into account the implications for the depending plans. These dependencies are further discussed in the next section.

In this research only the final output of the shaded part of the planning process in the above railway planning process, see Fig 3 above.

The three remaining problems are not directly connected to the demand, they are subject to the created timetables and hence left out of the scope of this study (they are also closer to the day of operation than the two first steps, i.e. strategic to operational). Detailed description, including the planning problems are discussed the following references **[23]**, **[36]** and **[41]**.

1.2 STATEMENT OF THE PROBLEM

In railway transportation system, train Scheduling is concerned with three major tasks namely train route assignment, train platforming and train timetabling as discussed briefly in the above section. Train scheduling is one of a challenging task for railway companies and it takes the major share for the success of these companies in both service quality realization and competitiveness. Though many researches have been conducted in this area there is no a general computer based solution available to solve different rail network problems. Further the challenges of train scheduling gets worse if the rail network size is large and complex. Scheduling by its nature requires experienced and well educated personnel like skilled rail network planners, skilled dispatchers to develop manual based schedule using train diagrams or other manual methods.

To the best of my knowledge, ERC has no such personnel currently since it is a beginner railway operator. These reasons together with my interest in programming initiated me to study a mathematical model and implementation of the corresponding optimization algorithm to solve train scheduling problem for Addis Ababa LRT system using computer program. Addis Ababa LRT has a total length of 34.25kms with 39 stations and two depots located at one terminal of the two corridors called East-West and North-South lines as show in the figures 1.4 and 1.5. The stations located in the intermediate are not used for train passing or shunting instead they are used to load and unload passengers. This LRT system is a double track system and the train movement is planned to be operated in one direction each track on the corridors.

In the previous research works of our senior student **[12]** the effect of disturbances that causes delay were not considered. Moreover connection, turnaround times and buffer times which can reduce the effect of disturbance are not considered. In this research the effect of delay is planned to be handled by using scheduling additional train services, SATS method. Increasing buffer time is one of the techniques to develop a robust scheduling which is resistant to the possible perturbations. By robust we mean less sensitive to disturbances. The parameters stated above will also be included in the model to be formulated.

1.3 RESEARCH OBJECTIVE

1.3.1 GENERAL OBJECTIVE

The main objective of this research work is to find a way to efficiently utilize the LRT infrastructure resources thereby reducing the waiting time of passengers and the effects of traffic delays on the operation of the LRT trains, and ensure conflict free operation of trains.

1.3.2 SPECIFIC OBJECTIVES

The specific objectives that have been studied in this research are listed below:

a. Describing railway planning stages giving especial focus for operational plans related to train scheduling and timetabling.
b. Present a suitable mathematical model to formulate train scheduling problem
c. Describe the algorithm that can be suited to solve the formulated problem
d. Develop train scheduling mathematical model for Addis Ababa LRT system
e. Design algorithm used to solve the scheduling problem
f. Implement the algorithm by deploying a computer program that generates optimized train schedule
g. The application developed is designed to handle the effect of perturbations due to fixed maintenance schedule and unplanned delays etc.

1.4 RESEARCH METHODOLOGY

The problem described above will be carried out by the following methodologies. In the first stage reference books, journal papers, conference reports related to train scheduling will be studied. In parallel to this data that are required as input for the research like design documents for Addis Ababa LRT system are collected. After acquiring these data and knowledge of scheduling the problem formulation (modeling) is developed. The model is then converted into algorithm that meets a realizable computational requirements such as acceptable computation time and memory space requirements. Finally the coding for the designed algorithm is written using visual C sharp computer programming language.

Eventually the scheduling software developed that way is used to generate train schedule for a set of railway vehicles available for operation. This schedule consists of trains arrival and departure times at each station and relevant locations line junctions and crossings including the routes to follow. Figure 1.6, shows the flow chart of the methodology used to carry out this research from start to end.

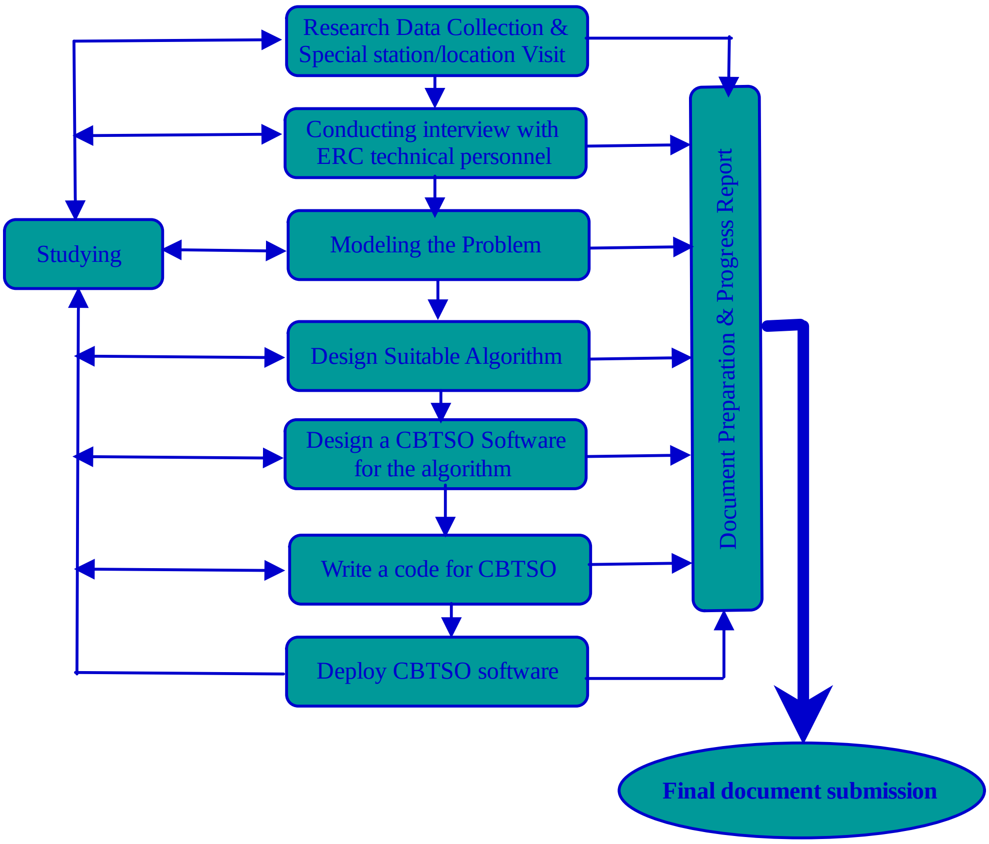


Figure 1.6: Flow chart of research methodology

The model used to formulate AA LRT passenger train scheduling problem is called constraint satisfaction programming and is presented in chapter 4 briefly.

1.5 SCOPE OF THE RESEARCH

The scope of Train Scheduling in this research paper is to develop a mathematical model and algorithm using the current state of arts in operations researches method for train schedule problem for light rail transit system. The algorithm show is then implemented for Addis Ababa LRT system using visual c# 4.0 computer programming language and Microsoft SQL Server database.

CHAPTER 2

2. LITERATURE REVIEW

As stated in the previous sections Train scheduling is one of the most challenging and difficult problems in railway planning processes and it is a very active part of railway researches. Railway companies in the world had been carrying out this task manually for more than a century through a trial and error process. There have been many studies of more efficient scheduling methods - simulation, mathematical programming, and expert systems (Knowledge based System) and so on.
The fundamental base of the train schedule is the single line plan which determines the number of trains serving the line connecting two terminal stations in a fixed time interval (Bussieck et al., 1997) **[31]**. Many researchers have tried to solve the realistic problems by adding various constraints and conditions on this basis.

Work on an optimum solution to the train scheduling problem started in the early seventies by Szpigel (1973) **[43]** who developed a linear programming model to determine the best overtaking and crossing positions given that the departure times and upper velocities of the trains are known. A Branch and Bound method is used to resolve the conflicts and lower bound to the remaining delay is generated by relaxing the remaining conflicts. Minimising the sum of the travel times was the objective and only small problems were tested. Petersen et al (1986) **[39]** considered a similar dispatch algorithm which calculates the crosses, segment transit times and determines which train takes the sidings in order to minimise the total travel times.

Kraft (1987) [**27**] takes a different approach by developing a dispatching rule giving the optimal time advantage for a particular train based on train priority, track running times and the delay penalties of each train. This rule was used to resolve crossing conflicts in the Branch and Bound procedure.

Kraay et al (1991) **[26]** proposes a model which paces trains in order to conserve fuel and, at the same time, keep the lateness of trains at a minimum. The fuel consumption is a function of friction and gradient of track, speed and mass of the train, and air friction. Two heuristics were proposed which are able to find solutions to realistic size problems. Jovanovic et al (1991) **[23]** uses a similar constraint framework as part of a decision support model called 'SCAN' which is based upon combinatorial optimisation and simulation. Mills et al (1991) **[30]** formulated a discrete network type model by discretizing the departure and arrival time variables. The discretization allows the use of the shortest path algorithm to update the journey of each train. A procedure recursively updates the path of each train until a feasible schedule is found. The solution procedure is an approximation.

Mees (1991) [33] models the single line rail as a network structure where each segment is an arc (a siding is considered as an extra arc), separated by nodes (considered as track intersections or stations). The network is time-space with a fixed schedule time span and headways are obtained by allowing only one train per arc at a time segment. A solution procedure similar to Mills et al (1991) [30] is used to find a feasible solution.

Morlok and Peterson (1970) [34] are known as one of the earliest work on an optimum solution to the train scheduling problem. The objective is to minimize the sum of fixed costs for trains, variable costs for transportation, handling and storage of freight, and opportunity costs of using rail equipment, while providing on-time deliveries of time-sensitive goods. Each potential train has a departure time, routing, set of stops, and an upper limit on cars. Decisions are which trains to operate and which freight to assign to each train. The authors apply branch-and-bound to solve a small instance of the resulting multi commodity network design problem.

Jovanovic and Harker (1991) [23] developed the SCAN-I model to construct timetable and pass plans with a focus on robustness against travel time randomness. They employed a branch and-bound variant called a process-interaction simulation to assess whether a timetable is feasible under deterministic assumptions. Whenever a feasible schedule is identified for the deterministic travel times, simulation is used to estimate the probability that the schedule is achievable for random travel times.

Kraay et al. (1991) [26] constructed MINLP problem for variable velocity while Jovanovic and Harker (1991) formulated a mixed integer problem for fixed velocity. The objective is to minimize train delays and fuel costs and resulting model permitted more flexibility in scheduling and fuel cost reductions.

Kraay and Harker (1995) [25] developed real-time scheduling model of freight railroads to provide a link between strategic schedules and line dispatching or CAD (Computer Assisted Dispatching) models. Carey and Lockwood (1995) [5] presented a timetabling problem on a single rail line assuming a constant velocity for each train. They constructed the timetable and schedule to minimize total weighted delay subject to maximize train velocity and siding length.

Higgins et al. (1996) [16] described the development of a model designed to optimize train schedules on single line rail corridors. The objective of the paper is to present a lower bound that will allow the branch and bound procedure to find the optimal solution to realistic size problems in reasonable time.

Marin and Salmeron (1996) **[32]** addressed an aggregate steady-state freight planning model in which train routes (including stops), their frequency, and the number of cars using each service are determined. Costs include a fixed charge for each train, handling and delay costs, and costs of investments in additional trains. Constraints are imposed on the number of cars transported on each track segment, the number of cars using each yard, and the number of trains. They suggest heuristics in which service frequency decisions are handled by simulated annealing or tabu search and freight routing is addressed using a network flow model.

Nozick and Morlok (1997) **[36]** addressed a finite-horizon, discrete-time problem of minimizing the total variable cost of moving loaded and empty trailers and flatcars given a fixed train schedule while satisfying due dates. They developed a procedure that involves iteratively solving a linear programming relaxation and rounding some of the resulting fractional values until a feasible integral solution is found.

Kwon et al. (1998) formulated a multi-commodity flow problem to determine car routes assuming the train schedule-blocking plan and block-to-train assignments are given. The goal is to minimize late delivery penalties while ensuring demand is met, cars are appropriately assigned to blocks, blocks are appropriately assigned to trains, and train capacity restrictions are enforced. They use a column generation approach to solve realistic problem instances. This model has proved useful for modifying train schedules when the initial train schedule does not provide adequate customer service.

Newman & Yano (2000) [**35]** presented both centralized and decentralized approaches for solving the discrete-time problem of simultaneously deciding train service on all possible nonstop links and freight allocation. They considered deterministic transit times and time-varying demand minimizing fixed and variable operating and storage costs.

Chang et al. (2000) **[6]** gave a multi-objective model for passenger train services planning. They determined the optimal allocation of passenger train services on an inter-city high-speed rail line without branches with specifying subset of stations at which the train must stop. This model belongs to the line-planning category of the hierarchical planning process.

Ghoseiri et al. (2004) **[13]** also developed a multi-objective model for the passenger train scheduling problem on a railroad network that includes single and multiple tracks and multiple platforms. The objective is to minimize the fuel consumption cost or total passenger time subject to the train movement continuity constraints, events continuity constraints and trip time and dwell time constraints and so on.

Ingo Arne Hansen and Jorn Pachl (2006) described how manual based scheduling was the typical main scheduling method used for more than 150 years on almost all railways in the world. Only the occurrence of high performance computers in the 1990s offered the opportunity to introduce computer based scheduling methods **[19]**. Manual based train scheduling requires skilled and experienced rail network planner. Further it requires a massive mental effort and long time to solve. The objective in this paper is to develop a computer based train schedule for Addis Ababa LRT system that can compute optimal or near optimal solution in reasonable time.

2.1. RAILWAY SYSTEM COMPONENTS AND TRAIN SCHEDULE

Understanding how train schedule depends on the available railway infrastructure is one of useful information used to clearly identify the constraints that are needed to be included in the model of scheduling problem. In its basic form a railway network is composed of stations, links and block sections separated by signals. For safety reasons, the signals control the train traffic on the routes, and impose minimum distance headway between consecutive trains. Signals, interlocking and Automatic Train Protection (ATP) control the train traffic by imposing a minimum safety separation between trains, setting up conflict-free routes and enforcing speed restrictions on running trains.

The signaling system of main line AA LRT is the key subsystem in the whole signaling system. According to the design principle of the Project, the signaling system of main line is divided into the following two zones by functions:

1. *Non-blocking protection zone:* The signaling system provides no space headway protection for the train in this zone. Instead, the driver drives the train through control of the running speed and the safety headway with the front train only by viewing the indication in the switch area signal.

2. *Blocking protection zone:* The signaling system provides space headway protection for the train in fixed blocking mode in this zone. The driver drives the train manually through control of the running speed by viewing the speed limit signs of the line and the indication of the signal. The definitions of signal displays to be used for Addis Ababa LRT system are shown on the figures in the following pages.

The minimum safety distance and time headways depend on the speeds of the consecutive trains, the braking rate of the second train, the train length of the first train and the signal spacing. In case of technical or human failures, ATP ensures safe rail operations. In particular, ATP causes automatic braking if the train ignores the valid speed restrictions. For the AA LRT system **signals** are located before every junction as well as along the lines and at both ends of each station. A **block section** (in a fixed block system) is a track segment between two main signals that governs the train movements, and may host at most one train at a time.

The railway signaling system deployed for Addis Ababa LRT system is characterized by the three-aspect fixed block signaling system. Its ATP system is called IATP system. This subsystem has main functions namely: stop the train before restrictive signal, over-speed protection; overrun red light protection, rollback protection, emergency brake control, train door supervision and PSR control according to the track.

AA LRT has fixed block signaling systems in which a train may enter a block section only after the train ahead has completely cleared the block section and is protected by a stop signal. The signal aspect used for AA LRT system is **red**, **yellow** or **green,** refer the figures 2.1 and 2.2.

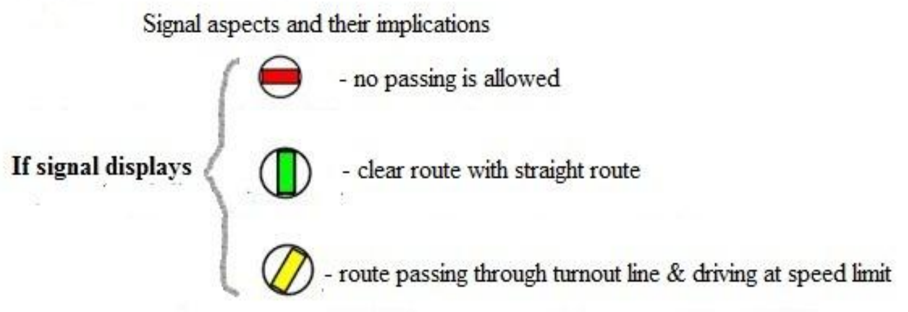


Figure 2.1: Description of AA_LRT system signal displays [7]

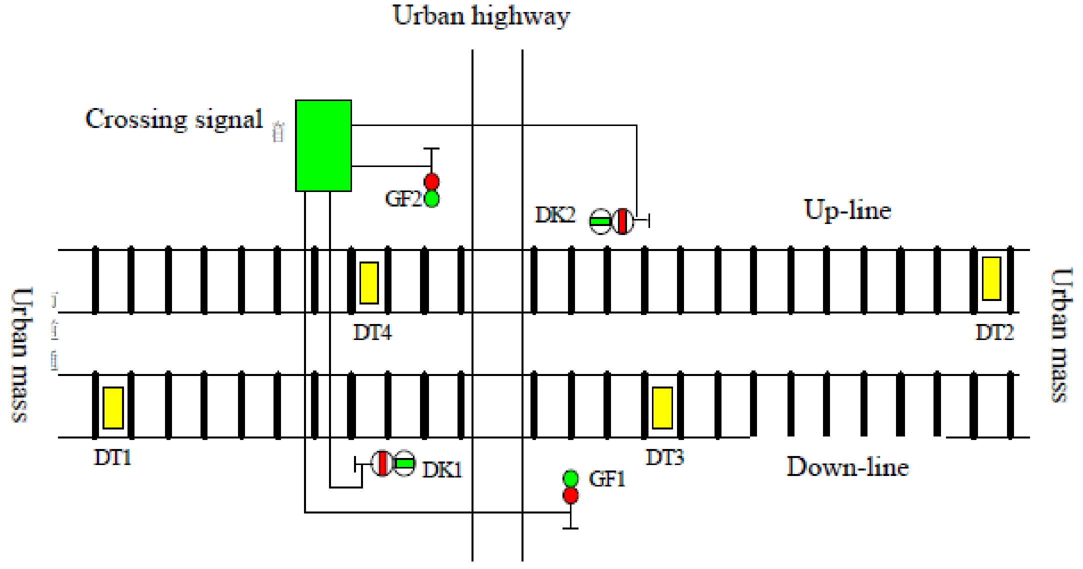


Figure 2.2 Signal system at level crossings of Addis Ababa LRT system [7]

A red signal aspect means that the subsequent block section is either out of service or occupied by another train, a yellow signal aspect means that the subsequent block section is empty, but the following block section is still occupied by another train, and a green signal aspect indicates that the next two block sections are empty. A train is allowed to enter the next block section if the signal aspect is either green or yellow, but the latter requires deceleration and stop before the next signal if this remains red. A detailed description of different aspects of railway signaling systems and traffic control regulations can be found e.g. in Pachl et.al. (2006) **[19]**.

The passage of a train through a particular block section is called an *operation*. A *route* of a train is a sequence of operations to be processed in a *track yard* during a *service* (train run). At any time a route is *passable* if all its block sections are available and the corresponding block signal is green or yellow, i.e., there are no blocked tracks. The *timing* of a route specifies the starting time t_i of each operation in the route. Each operation requires a traveling time, called *running time*, which depends on the actual speed profile followed by the train while traversing the block section. A speed profile is furthermore constrained by *the rolling stock characteristics* (maximum speed, acceleration and braking rates), *physical infrastructure characteristics* (maximum allowed speed and signaling system) and driver behavior (coasting, braking and acceleration profiles when approaching variable signals aspects. The running time includes the time needed to accelerate (or decelerate) a train due to a scheduled stop, as well as speed variations between two consecutive speed signs. Furthermore, the *running time is known in advance since all trains travel at their scheduled speed, which usually contains some margins for recovery*. In ***yards***, the routes for the individual trains need to be setup before entering and cleared after leaving, which takes a certain switching time.

A delay may occur when a train reaches the end of a block section and the subsequent block section is still occupied by another train. The running time of a train on a block section starts when its head (the first axle) enters the block section. Safety regulations impose a minimum distance separation among the trains, which translates into a minimum *setup time* (time headway) between the exit of a train from a block section and the entrance of the subsequent train into the same block section. This time takes into account the time between the entrance of the train head in a block section and the exit of its tail (the last axle) from the previous one, plus additional time margins to release the occupied route and sighting distance (as described e.g. in Hansen(2006) **[19]**. Railway timetable design usually includes *recovery times* and *buffer times* between the train routes: recovery times can be utilized to recover from delays by running at maximum speed and shorten the scheduled train stops to a minimum dwell time, whereas buffer times prevent or reduce the delay propagation to other trains.

In this research a cyclic timetable is deployed which describes the movement of all trains circulating in the network during subsequent time periods, specifying, for each train, the planned arrival and departure times at stations. At stations, a train is not allowed to depart from a platform stop before its scheduled departure time and is considered late if arriving at the platform later than its scheduled arrival time. At a platform stop, the scheduled stopping time of each train is called dwell time. Additional practical constraints related to passenger satisfaction are to be taken into account, such as minimum transfer times between connected train services. This is the time required to allow passengers alight from one train, move to another platform track and board the other train.

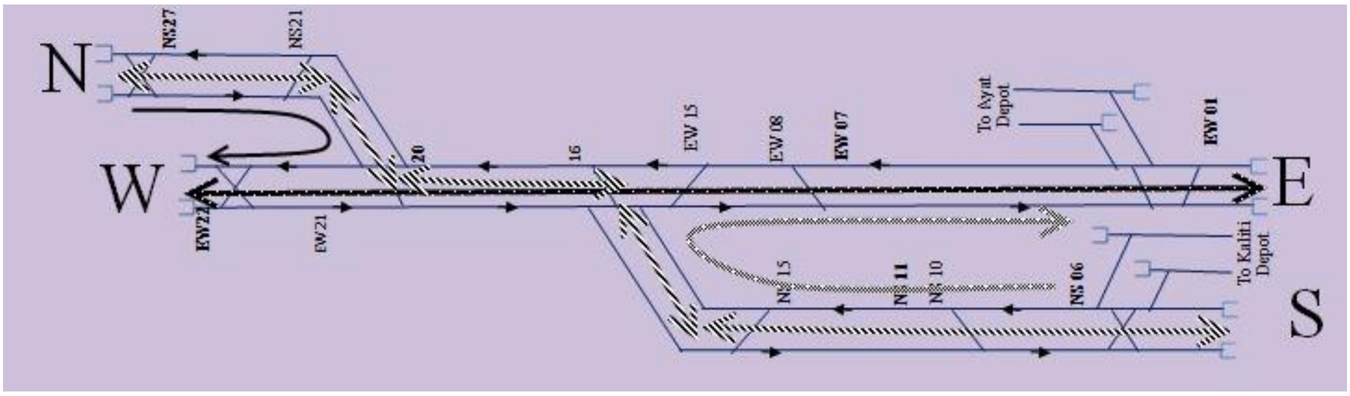


Figure 2.3: AA LRT Train routing plan and connection time constraint graph

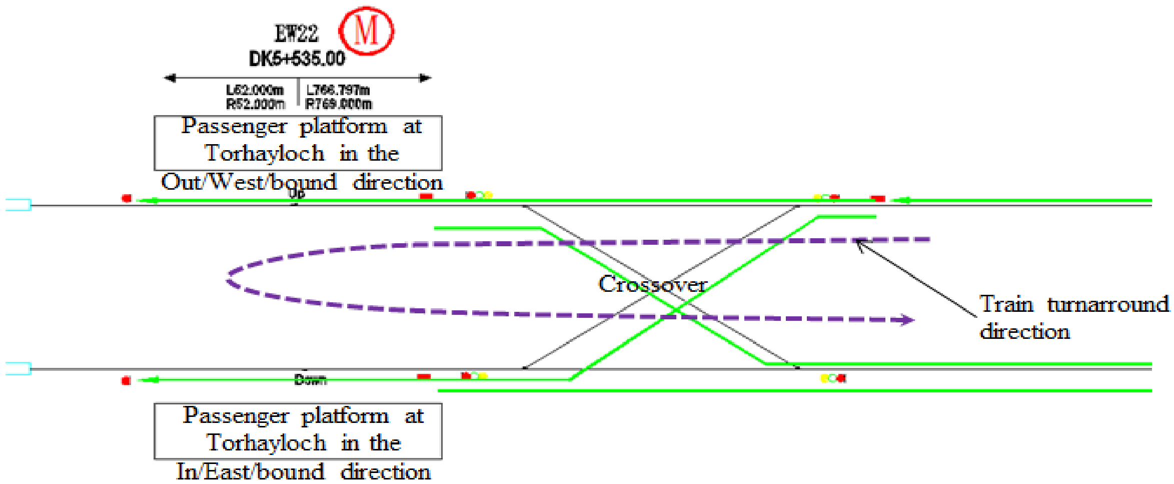
Key:

passenger transfer path from North end to West end

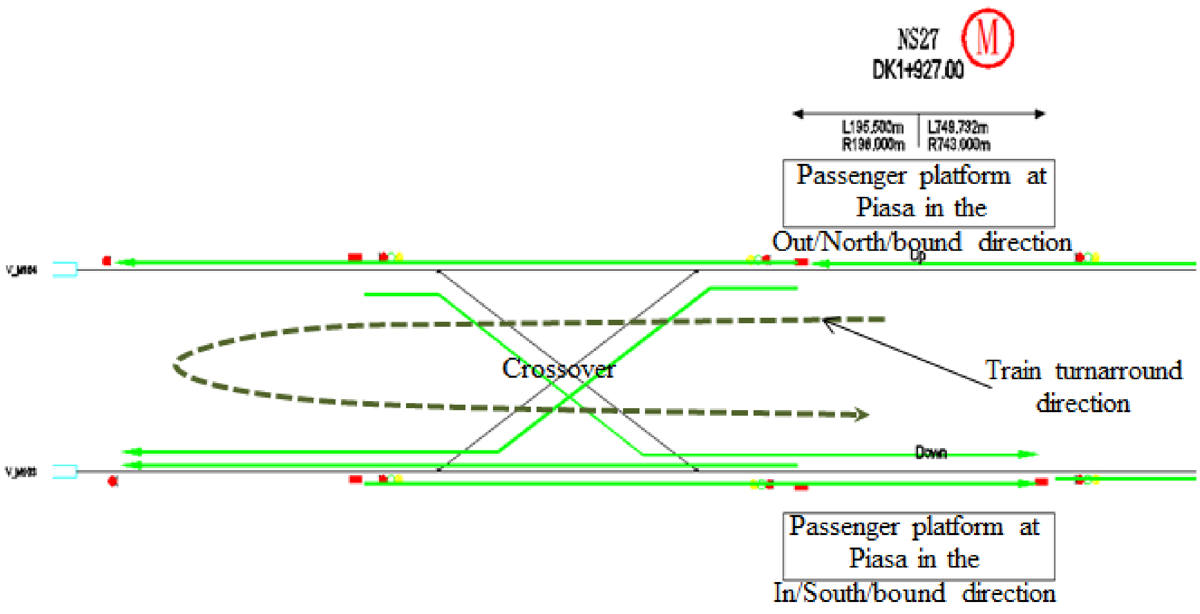
passenger transfer path from East end to South end

East-West Main route

Route changing locations for passengers: Sstadium and Lideta



(a)



(b)

Figure 2.4: (a) & (b) shows cross overs located at EW and NS terminal points respectively

Constraints due to rolling stock circulation must also be taken into account. In fact, a train completes a number of round-trips during the service of a line and its length may be changed by (de)coupling. For short term plan AA LRT system uses one train unit with three cars with a total length of 30meters and two train units in the long term plan [7]. Hence, there is coupling (decoupling) operation will be carried out in a daily basis at least in the short term plan. In spite of these operations, railway timetables include a ***turn-around time*** at terminal station, which is a time margin between the arrival of the train and the start of a new train service in the opposite direction using the same rolling stock. In case of severe disturbances a scheduled train may be delayed from the beginning because of the unavailability of rolling stock or train personnel.

CHAPTER 3

3. TRAIN SCHEDULING PROBLEM

Railway systems are packed full of challenging combinatorial optimization problems. Yet, due to their complexity, the optimization process is usually carried out in sequence, subdividing the full problem into several sub-problems, which are solved one after the other (i.e., the output of one problem becomes the input of the following one). These sub-problems are NP hard and include (in order): Line Planning, Train Timetabling, Train Platforming, Rolling Stock Circulation, Train Unit Shunting and Crew Planning. Detailed planning stages of these sub problems are discussed briefly in the previous section 1.1.3 (planning stages of railway transport).This research work concentrates on two phases, namely Train Route Allocation Problem (TRAP) and Train Timetabling Problem (TTP). These two phases come right after the Line Planning Problem; hence they assume that the routes for the trains as well as the types and frequencies of the trains on each route have been defined. TRAP and TTP are solved in sequence. The TRAP is solved for a set of trains such that their specific operational route is allocated. Then, TTP is solved for a set of trains on a given railway route to determine a feasible schedule. Note that since all the stations in Addis Ababa LRT system has no platform for train parking other than the main track connecting each adjacent station, so there is no TPP (Train Platforming problem) problem we need to solve here.

3.1 ADDIS ABABA LRT SYSTEM DESCRIPTION

The Addis Ababa Light Rail Transit (AA-LRT) Project is composed of two routes called East-West Line which connects Hayat depot with the terminal point at Torhayloch (EW22) and North-South Line connects Kality depot with the terminal station (NS27) at Minilik II square near St George church. The trains are modern and they are 70% low-floor trams. Primarily, ground lines are established, but in some sections, elevated or underground lines are employed. The Project is a semi-closed urban rail transit system. The LRT vehicles pass some intersections in the form of level crossing. The signal system is designed on the principle that the LRT vehicles shall pass intersections first.

The planned lines are 75km in the overall length, and the lines in the E-W & N-S （Phase I Project to be executed currently is approximately 31.3km in length, including a common-track section of 2.63km which extends from stadium to Lideta station. Starting at the mileage of YCK5+046 and ending at the mileage of YCK22+464, the main line of the East-West Line is nearly 17.410km long, while the main line of the North-South Line is 16.561km long, with the starting mileage of YCK1+822 and the ending mileage of YCK18+381**[7]**.

At the end of the LRT 1st project phase there will be a total of 39 stations of which five of them are located on the shared line (common track section). There are equal number of 17 stations on both EW line and NS line on the track sections other than the shared line.

For the East-West Line, a parking lot is reserved near the starting point (not build in current stage) and the Ayat depot is built near the ending point; for the North-South Line, a parking lot is also reserved at the starting point (not build in current stage) and the Kality depot is built near the ending point. The parking lots reserved for the long term will not be constructed in the short term.

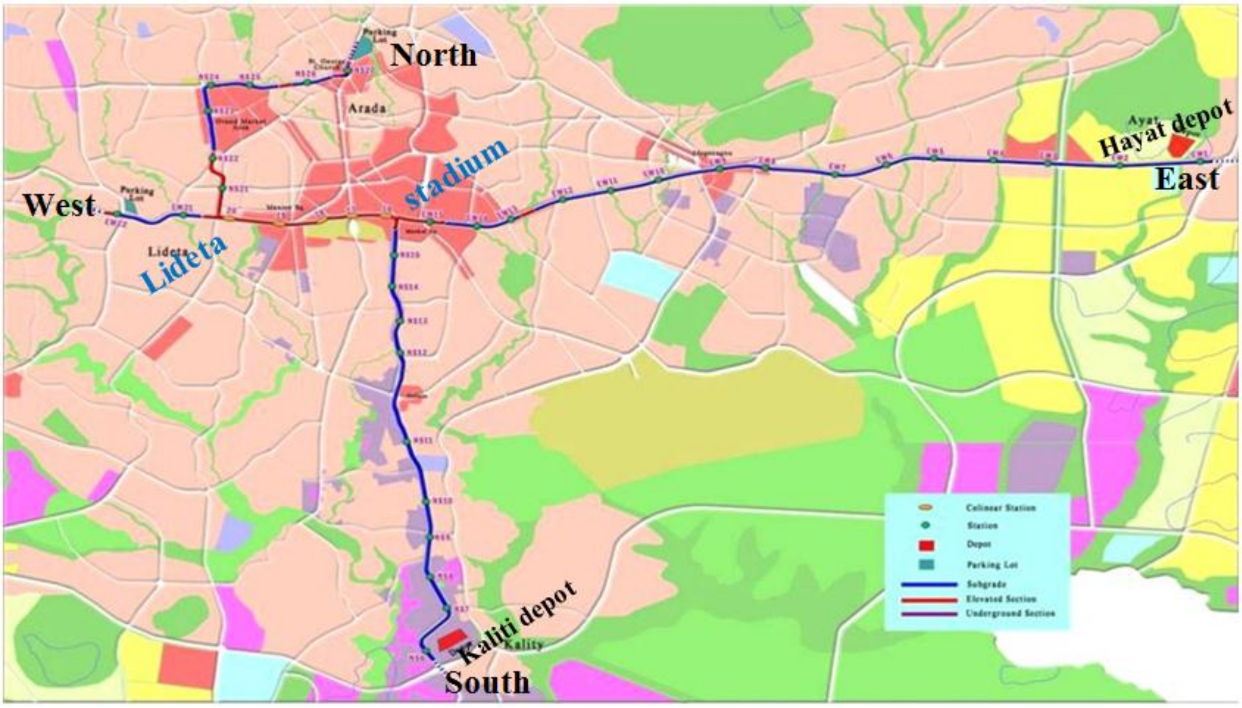


Figure 3.1: Addis LRT system topological drawing [7]

3.1.1. AA LRT ROUTES

Phase I project of the LRT system includes EW Line and NS Line, and the total length of the main line is about 31.3km. The total length of the main line of EW Line is about 17.410km. The EW Line and NS Line use a common track of about 2.63km from the intersection of Chad St. to the west of Meskel square and 5 elevated stations are arranged for the common track section.

There are two depots one for each line, Ayat depot for EW line and Kality depot for NS line. The EW Line (Phase I) starts in the west from the west of the loop line nearby Tor Hailtoch hospital, and pass through Mexico Square, Mesekel Square and Megenagna until Ayat. Totally, there are 22 stations, wherein, 6 elevated stations (including 5 common-track stations), and others are ground station. The maximum station spacing is 1.26km, the minimum station spacing is 0.5km, and the average station spacing is 0.813km. Whereas the NS Line (Phase I) starts in the north from the east of St. George Church, change the direction to the south after a section of Mercato market in the north, go through the west of this market until Chad Street and go to the east after being connected with EW Line, and change the direction to the south after reaching the west of Mesekel Square, and then pass under Meshalokia Bridge, China-Ethiopia Friendship Bridge until Kallti. Totally, there are 22 stations on this line, wherein, 7 elevated stations shall be arranged (including 5 common-track stations）, 1 underground station, and others are ground station. The maximum station spacing is 1.781km, the minimum station is 0.5km, and the average station is 0.771km. Refer the following table for the summary of this topic.

Addis Ababa LRT stations and Depots		
Station type	Number of stations	Remark
Elevated stations	9	5 on the shared line, 1 on EW line,3 on NS line
Underground stations	2	State here the location
Semi underground stations	1	State here its location and route
Ground stations	27	13 on NS line and 14 on the EW line
Total number of stations	**39**	State here the # of stations on each route
Depot	Location of the depot	Remark
Kalitiy (main control center for	On south end of the NS	main control center for the LRT system
Hayat	On East end of the EW	

Table 3.1: Addis Ababa LRT stations and depots summary table

3.1.2. INITIAL STAGE & LONG TERM ROUTING PLAN OF AA LRT SYSTEM

EW & NS lines adopt unidirectional train routing with a total of 39 stations of which 5 of them lie on the 2.5km common track section and an average traveling speed of 20km/hr on the whole line.

Both EW & NS lines adopt a unidirectional train routing with a total of 5 stations on the 2.5km long common track section and an average traveling speed of 20km/hr on the whole line. You can refer figures 3.2 to 3.4.

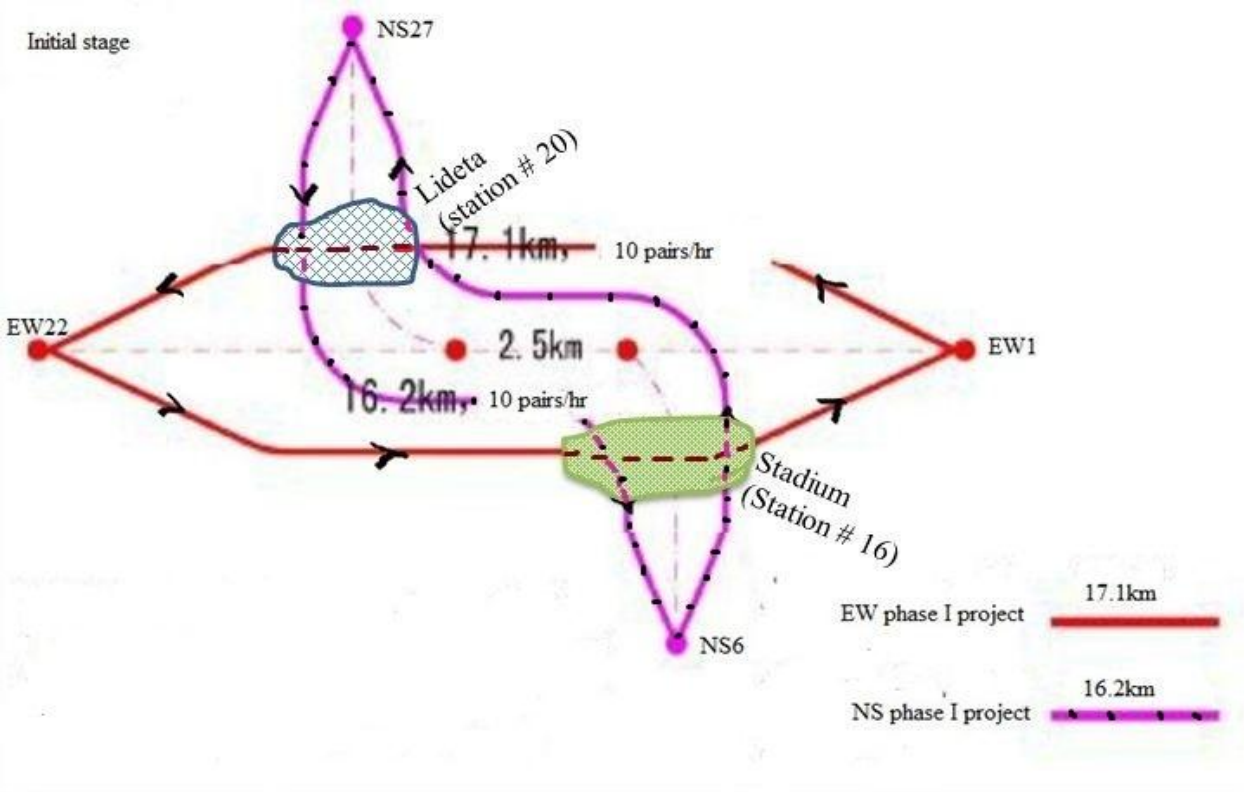


Figure 3.2: Initial stage routing plan of AA LRT system

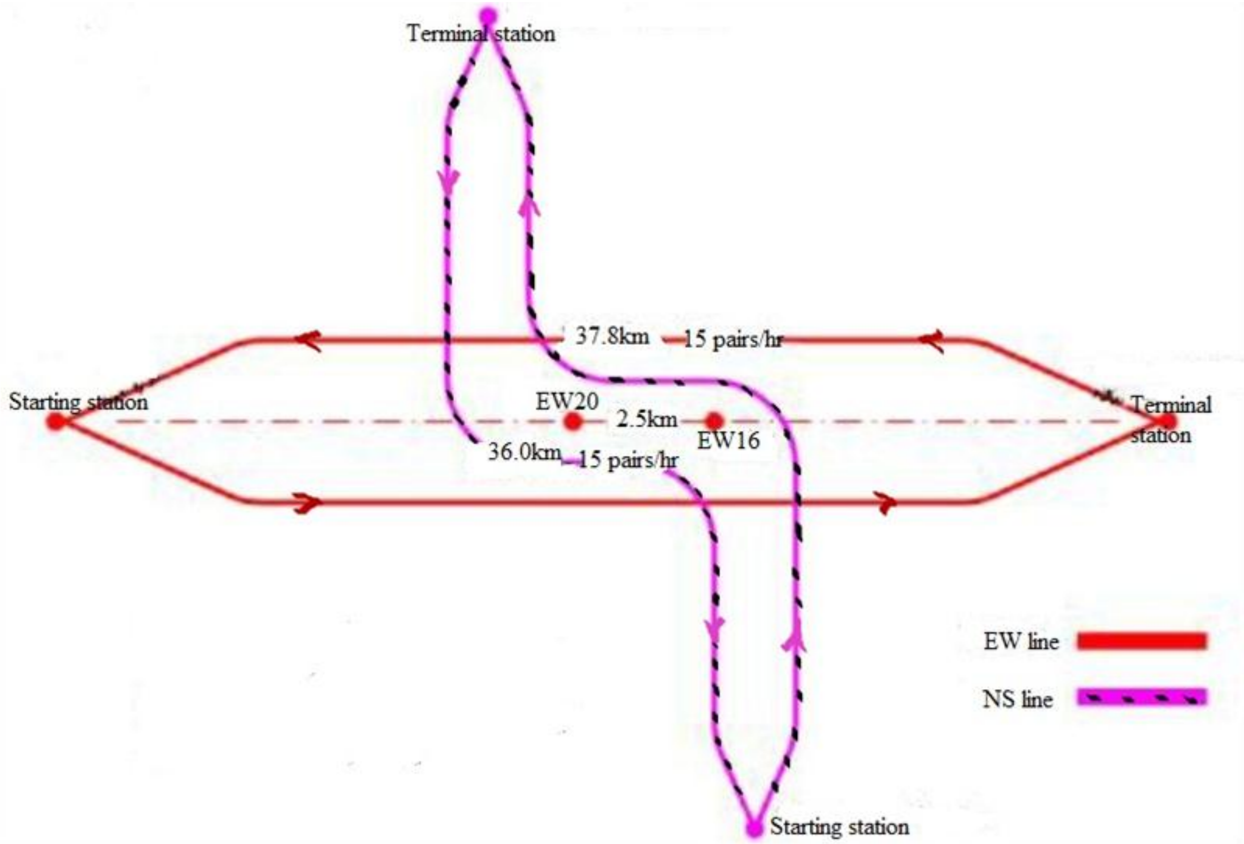


Figure 3.3 long term stage train routing plan of AA LRT system

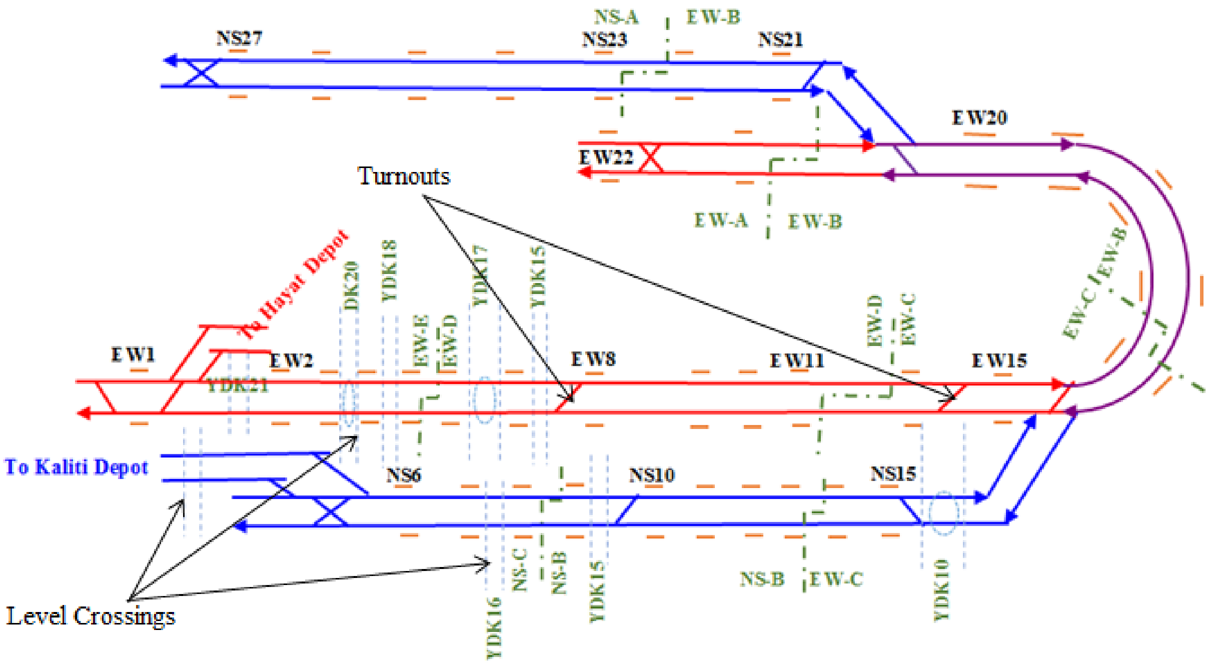


Figure 3.4: Rough LRT train routing schematic diagram

3.1.1. DAILY TRAIN OPERATION PLAN

The following two tables show the number of train pairs involved for daily operations for different time intervals on the two routes.

Daily Operation Plan of East-West Line		
Time interval/plan	Initial stage (number of train pairs/hr)	Long term (number of train pairs)
5:00~6:00	6	8
6:00~7:00	8	10
7:00~9:00	10	15
9:00~17:00	8	12
17:00~19:00	10	15
19:00~22:00	8	10
22:00~23:00	6	8
Total (train pairs /day)	**148**	**212**

Table 3.2: Daily operation plan of East-West Line

Daily Operation Plan of North-South Line continued		
Time interval/plan	Initial stage (number of train pairs/hr)	Long term (number of train pairs/hr)
5:00~6:00	6	8
6:00~7:00	8	10
7:00~9:00	10	15
9:00~17:00	8	12
17:00~19:00	10	15
19:00~22:00	8	10
22:00~23:00	6	8

Total (pair /day)	148	212

North south/East West train operation per day	
Number of train allocated per hour for operation	operation hours
10/10	Morning peak hour: 7:00 to 9:00 (2hrs)
	Evening peak hour: 17:00 to 19:00 (2hrs)
8/10	Morning Normal working hour: 6:00 to 7:00 and 9:00 to 12:00 (4hrs)
	Evening Normal working hour: 12:00 to 17:00 and 19:00 to 22:00 (8hrs)
6/6	Morning off peak hour: 5:00 to 6:00 (1hr)
	Evening off peak hour: 22:00 to 23:00 (1hr)
total number of trains allocated/18hrs = 148/148	Total working hour: 18hrs

Table 3.3 Daily operation plan of North-South Line

3.1.2. AA LRT STATIONS

Addis Ababa LRT system has station capacity of two trains, i.e. one train one each track. The 3D view of a typical LRT station is shown in the figure 3.5.

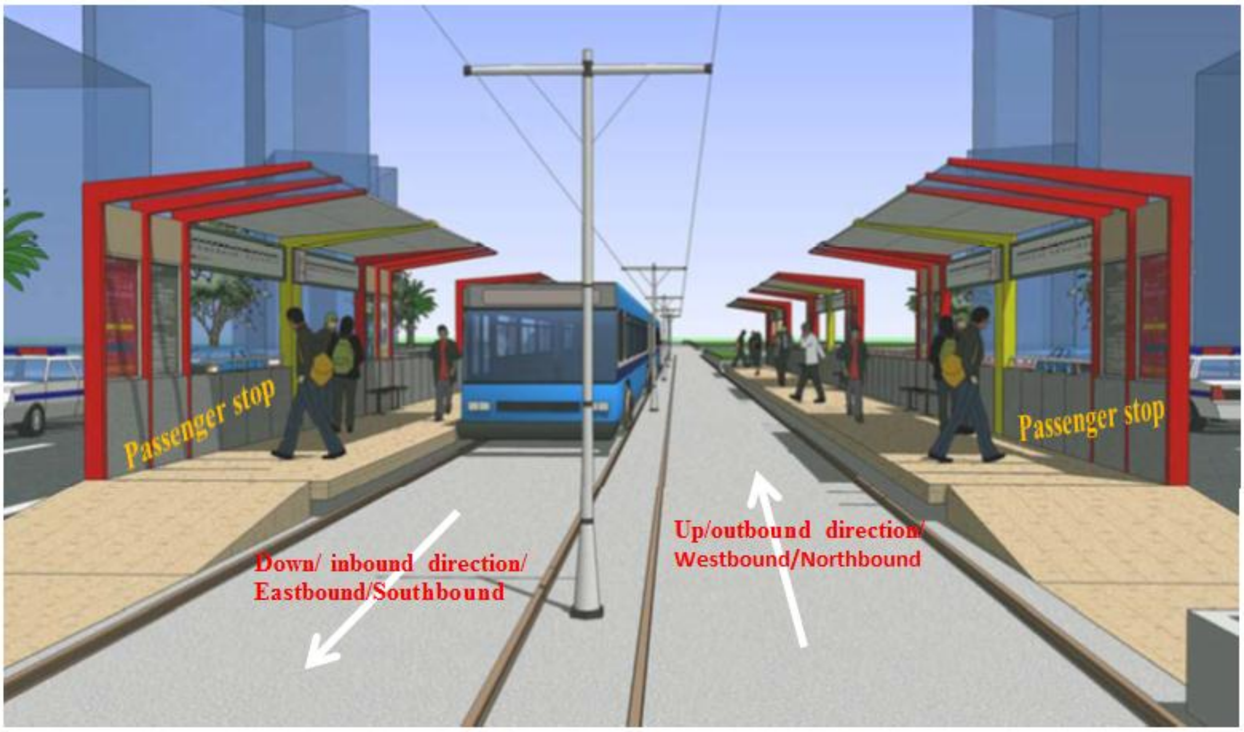


Figure 3.5: 3D view of typical AA LRT station

3.1.3. AUXILIARY LINES

The main type of auxiliary lines is single crossover, which performs different functions in different stations. The four detailed functions are as follows:

1. **Reversing Station**: single crossover is placed behind reversing stations for the trains to reverse directions. Refer fig __cross over and components of rail network.
2. **Depot Junction**: the scheme of using two rails to connect depot is adopted. The two rails (double track) directly extend from the terminal to depot.
3. **Starting Station in the Shared-Rail Section**: single crossover is placed to ensure two trains travel within single rail.
4. A few of single crossovers are placed in intermediate station for the trains to temporarily reserve directions.

See Table 3.4-1 Schematic Diagram for Detailed Arrangement of Auxiliary

ADDIS ABABA LRT SYSTEM TRAMCAR OPERATIONAL CONDITIONS

Safeguarding Measurement

1. The Section with Shared Rail: Interlocking system is adopted in the section with shared rail in order to ensure the operation safety in the intersection point.
2. Other Sections: Driver's operation ensures the security and safety of the tramcars.

BASIC TECHNICAL CONDITIONS OF AA LRT VEHICLES

Tramcar Type

1. After the execution of project I Addis Ababa LRT will use 6-axle double-articulated 70% low-floor Light Rail tramcar. The vehicles are 70% low-floor articulated 6-axle modern trams, consisting of three modules, bi-directional driving. Two tramcars will be able to operate with double heading. See fig 3.6 for one train unit which is going to serve A-LRT for the first term plan of the LRT system.
2. One unit of the tramcar is formed with three modules, which are called Mc, T and Mc. Under the Mc module of the tramcar body a bi-axial power bogie is installed, and a bi-axial driven bogie with independent wheel is installed under the T module.

Figure 3.6: One unit of AA LRT tramcar configuration

Marshalling Scheme

1. **Initial Stage**: one unit of the tramcar is arranged to be operated. The rated passengers are 286 persons in one tramcar （with 64 seats and 6 standing persons/m^2）.
2. **Long-term**: two units are coupled together to form one train. The rated passengers of the train are 572 persons （with 128 seats and 6 standing persons/m^2）

Vehicle type and train formation

Train formation: –Mc+Tp+Mc–

Mc module: motor car with driver's cab

Tp module: trailer without driver's cab and with pantograph

+: articulation device

-: Hidden folding coupler

Main dimensions of vehicle

Length of car body: ≈30m

Side doors of passenger compartment: four pairs per side

Clear opening of passenger compartment door (width x height): ≥1300×1860 mm

Seating capacity of vehicles

Number of passengers (persons)	Seated	Standing	Total
Seats (AW_1⇒ fully seated or standing: 0 person/ m^2)	65	0	65
Seating capacity (AW_2⇒standing: 6 persons/m^2)	65	189	254
Overload capacity (AW_3⇒standing: 8 persons/m^2)	65	252	317

Table 3.4: LRT vehicle loading capacity

LRT train speed

Maximum operation speed: 70 km/h

Average travelling speed: ≥ 20km/h (average dwelling time of **30** seconds at each station)

Operation speed during car wash: 3~4 km/h

Average acceleration:

Under rated load and rated voltage on straight and dry track, with half-worn wheels, average acceleration is as below:

Vehicle speed from 0km/h to 40 km/h ≥1m/s^2

Vehicle speed from 0km/h to 70 km/h ≥0.5m/s2

Average braking deceleration:

Under rated load on straight and dry track, with half-worn wheels, average deceleration from maximum vehicle operation speed of 70 km/h until stop is as below:

Maximum service brake deceleration: ≥1.1m/s2

Emergency brake deceleration: ≥2.0m/s2

3.2 INPUT FOR THE PROBLEM

- Number of stations
- Routes and direction of routes
- Dwell time at stations
- Turnaround time at terminal station
- Daily starting and ending time of trains operation

CHAPTER 4

4. MATHEMATICAL FORMULATION OF THE PROBLEM

4.1 INTRODUCTION

A way to represent and describe a problem clearly and easily has been a subject of intense research for many years. The result of this research effort is that many problems in the real world can be represented by constraints, and the satisfaction of these constraints gives solutions for the respectively represented problems **[48]**.

In the computer science area, the Constraint Satisfaction Problem (CSP) paradigm can be tracked back to the research in Artificial Intelligence and Computer Graphics in the sixties and seventies which focused on explicitly representing and manipulating constraints in computational systems **[49]**.

However the ease of representing a problem using constraints did not bring the same ease level of resolving the problems. This is because CSPs are combinatorial in nature, and moreover most problems of interest are NP-complete, therefore an efficient algorithm is unlikely to exist.

Until recently, the implementation of algorithms for solving CSPs in software tools, Constraint Programming (CP) languages, was not powerful enough to call the attention of the practitioners. The paradigm was thus confined within academic circles. However, this has been changed since the last decade. The CP paradigm has been attaining more support not only from the purely research circles but also from the commercial applications sector. This growth has been boosted by the existence of tools such as: Oz, CHIP, ILOG Solver, and ILOG Scheduler, which have provided practitioners with means of devising successful large real world applications. Some of these applications are cited in this thesis.

The successful of these CP tools is due to a huge effort in research over the last decades. The fruitful results of these years of research are now incorporated within these tools as inference mechanisms for transforming a problem into another equivalent, but simpler problem to be solved.

This development has greatly improved the CP tools on solving a variety of classes of real world problems. Furthermore, CP tools provide the user with a very expensive framework for dealing with constraints as they can be either linear, nonlinear, or both. Some other tools such as those for Mixed Integer Linear Programming (MIP) for instance, restrict the user to use only linear constraints.

The following sections give an overview of the main aspects and technics existent in the CP area. In addition, our attention is to present some of the characteristics of CP which make this paradigm rather different from the other techniques, for instance MIP in Operational Research. Given a CSP representation of a problem, we might want to find a solution for the system of constraints, or to find all possible solutions, or to find the best (optimal) solution, when criteria are given to measure the quality of solutions.

Many of the modern programming languages incorporate Object Orientation (OO), and this can be taken into the CP languages. Thereby, the sort of object structures that CP can manipulate range from simply numerical variables to endless number of complex structures. Although this modeling method is used here for scheduling, CP language can be applied for different class of problems.

4.2 DEFINING A PROBLEM AS A CSP

In the CP paradigm the emphasis lays in the manipulation of the variable domains and the relations among these variables- the constraints.

A CSP can formally be defined as a triple: (V, D, C), where V is the finite set of variables in the $\{v_1, v_2, \dots, v_n\}$ to represent the problem.

D is a function which maps every variable in V to a set of objects of arbitrary type. Hence, $D(V_i) = D_i$ is the domain of variableV_i. In this research we will only consider variables with integer domains, although a CSP can also be defined with other type of values for the variable domains.

C is the set of constraints which restrict the values that the variables can simultaneously take. A constraint $c_i \epsilon\ C$, between variables $v_i, v_j, v_k, \dots$, is any subset of possible combinations of values of these variables, in other words $c_i(v_i, v_j, v_k, \dots) \subseteq (D_i \times D_j \times D_k \times \dots$ [50]. However, the constraints of real problems are not represented this way in practice. This extensional way of representing a constraint may be, if not impractical, so computer memory-consuming.

Another way of representing constraints is intentionally when constraints are expressed as a mathematical relationship the set of allowable n-tuples is implicitly defined. The model used in this research work refers to this part of CSP. See the diagram shown on figure 4.1 how CP is a popular and a powerful tool to model different types of real world problems that involves a variety of complexity.

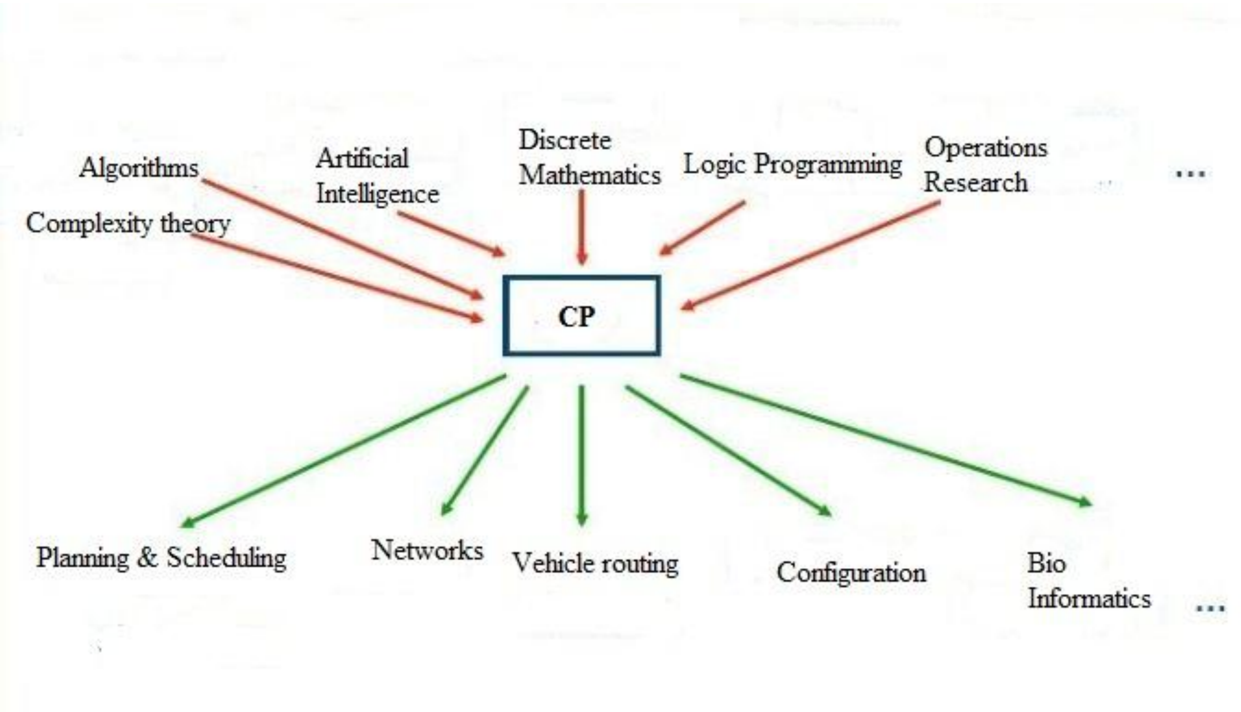


Figure 4.1: Advantages of constraint Programming model

In train scheduling problem once the infrastructure and the rolling stock are defined, in the second stage of railway line definition the operational policies should be determined. First, different train services should be defined. Each train service is associated to a determined route (topological path), a set of planned stops (every train of the route stops at the same stations) and to a determined kind of train. The operational pattern of the line should be defined (mix of trains, frequencies, etc.) as well as the boundary conditions. Finally, other operational constraints can be considered (correspondences, etc.).

The railway system has been modelled according to that structure. The information required and the objectives at each step are the following:

Step 1: Topology of the line

The main objective is to model the line splitting it into small standardized sections; nodes (which refers to stations, terminals and junctions which are defined below) and connections between them. The choice of these standardized sections is a generalization of the sections proposed in (Pachl 2002; Landex 2009). The connections between the nodes are the inter-stations.

There are seven kinds of nodes required to model any railway topology:

N1. **Terminal stations:** Final station of a line. Every train must stop and turnaround. For Addis Ababa LRT system we have two terminal stations where there is a cross over track so that the train can turn around to change their track and movement direction. These terminals are located at one end of each corridor. The terminal on the EW line is positioned at Torhayloch and for NS line it is placed at Piasa/Minilik II square near St George church.

N2. **Turn-around stations:** Intermediate stations of a rail line where trains can either turn-around or continue travelling in the same direction. As it can be seen from fig ----- there are 12 turnouts/turnaround stations.

N3. **Stations**: sections of the line where the trains might stop.

N4. **Overtakes:** sections of the line with one track where a train can be stopped till other train travelling in the opposite direction or even a faster train travelling in the same direction overtakes it. This node is not available for AA LRT system since the system is a double track and if overtaking of train is needed it can use the other track as overtaking track. But this is not the case in AA LRT operational rule.

N5. **Junctions:** sections of the line connecting different inter-stations.

N6. **Crossovers:** sections of the line where trains can change from one track to another.

N7. **Number-of-track change points:** Sections of the line connecting two interstations with different number of tracks.

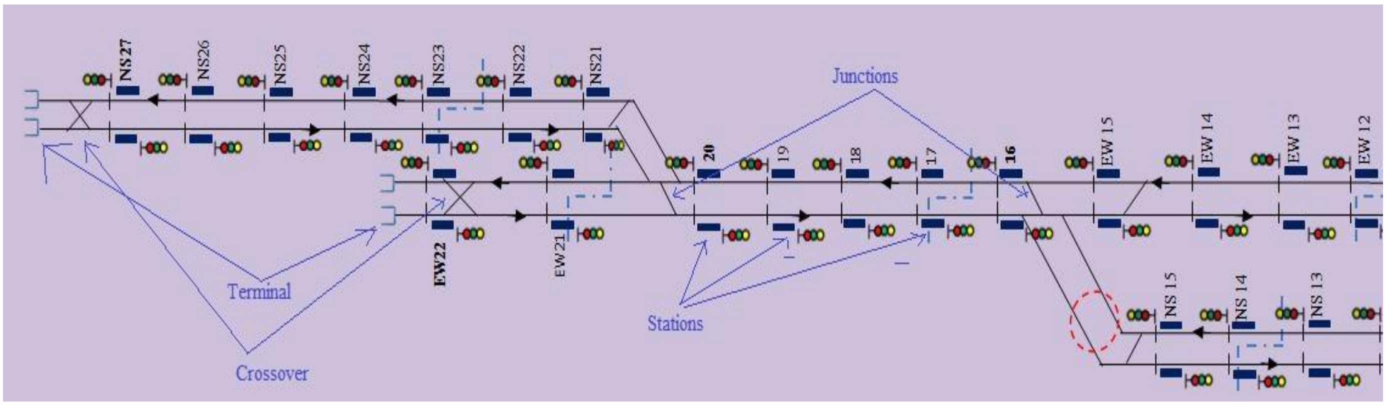


Figure 4.2: Types of nodes connected with inter-stations

Step 2: Signalling system

The objective is to determine pairwise incompatible itineraries at each section of the line. The information required at this step of the model is minimum safety headways for each kind of train travelling on the line, for each kind of event (arrivals, departures, passes) associated to each pair of pairwise incompatible itineraries.

Step 3: Train service

Train services are determined by:

- a commercial service (defined by a topologic path and by a set of planned stops)
- the kind of train offering the service
- the possible tracks of each section that these trains can use Minimum and maximum stopping times and travel times are defined at this step

Step 4: Demand

The objective of this step is to determine the number of trains offering some train service during the design time window (the frequency of each service) to meet a determined demand. Additional information required to schedule the trains is usually given in terms of:

- Operation patterns at some sections: defined by the order of events of the trains in a section during a determined period of time (time window).
- Other ways, as for example, minimum and maximum headway between two consecutive trains accomplishing the same train service in some sections of the line.

Step 5: Boundary conditions

Some information about the position of the trains along the line at the beginning and at the end of the time window is also required. This data can be provided

- Defining the initial position of the trains at the beginning of the time window.
- Scheduling the first event of each train scheduled.
- Using periodic operation policies.
- Other ways, as for example, minimum and maximum headway between two consecutive trains accomplishing the same train service.

The link between one service of the train and the following one should also be defined. Dwell times for preparing the train for a new service should be considered. The possibility of having trains accomplishing different train services is decided at this step.

CHAPTER 5

5.1 MATHEMATICAL MODEL

The mathematical model formulates the topology of the line as a graph whose vertices represent the nodes and the arcs represent the connections. Each vertex and each arc have associated the number of tracks. The vertices have associated also the kind of vertex. The stations are modeled as successions of tracks of nodes or connections. The routes can be modeled as a tuple of an itinerary, a kind of train, a determined plan of nodes where the train stops and a determined plan of nodes between which the travel time for the train should be determined. The information of the signaling system and of the operational policy is used to formulate the constraints of the model.

Before going directly into problem formulation let's present the assumption made for the formulation.

5.2 MODEL FORMULATION

Assumptions

This paper considers unidirectional operation of an urban transit line consisting of J stations, where station 1 is the origin station and station J is the final station of each trip. Before formulating the real-time scheduling problem, the following assumptions are made for Addis Ababa LRT system:

- The track is divided into different length of segments which are separated by **intermediate stations**. See table appendix for (distance between stations)
- Trains are assumed to stop at each station while moving from route starting point to the route terminal.
- No overtaking is allowed along the tracks. The operation is first in first out type since there is no siding track at stations.
- Trains can follow each other on a track segment with a minimum headway. For double track sections, it is assumed one lane will be allocated for inbound trains and one will be allocated for outbound trains. Usually, signal points will be set up this way.
- Scheduled stops are permitted at any intermediate stations.

5.2.1 PARAMETERS OF THE PROBLEM

The following are the parameters used in CBTSO model:

- Trip number
- Frequency / number of trains assigned per hour per route
- Starting and ending time of the operational times
- Line/route
- Route directions
- List here the remaining parameter whenever you find them helpful to be included in your train scheduler

To calculate the number of trains that can be operated on each line during a single period, sue the following railway line capacity formula.

$$Railway\ line\ capcity\ estimation\ formula = C = \frac{Period\ of\ operion\ in\ minutes}{headyway\ time\ in\ minutes} \ldots\ldots\ldots\ldots\ldots\ldots\ldots .1$$

For Addis Ababa LRT system the designed safety headway time between two consecutive trains is 6minutes. Let us use one hour as a fundamental train scheduling period. Then the number of trains that can be scheduled will be calculated us:

$$Railway\ line\ capcity\ = C = \frac{60min}{6min} = 10\ trains/hr$$

This figure indicates the number of train that can be scheduled on a single LRT line in one direction only, i.e. we will have 20 trains that can serve on a single line for both directions, up/outbound and down/inbound.

5.2.2 VARIABLES OF THE PROBLEM

Describe the variables as binary and integer and continuous variables

Decision Variables: Variables X's: A variableX^{Δ_i,∇_j}determines which train (Δ_i or ∇_j) arrives earlier to the crossing station. If X^{Δ_i,∇_j} = 0, then train T_i arrives at the crossing station first and train T_j arrives at the same station later:

$$X^{\Delta_i,\nabla_j} = \begin{cases} 0, \text{if train } \Delta_i \text{ arrives at the crossing station first} \\ 1, \text{ if train } \nabla_j \text{ arrives at the crossing station first} \end{cases} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 2$$

You may refer fig 4.4 for the crossing points available in the LRT system.

a. **Variables *Y's*:** A variable $Y^{\Delta_i,\nabla_j}_{k\leftrightarrow k+1}$ determines the track between station k and station $k+1$ in which train Δ_i crosses with train ∇_j. If $Y^{\Delta_i,\nabla_j}_{h-1\to h} = 1$ for $h \leq k$ and $Y^{\Delta_i,\nabla_j}_{p\to p+1} = 0$ for p $\geqslant$ k then, the crossing between train Δ_i and train ∇_j is carried out in station k:

$$Y^{\Delta_i,\nabla_j}_{k\leftrightarrow k+1} = \begin{cases} 1,\ \text{for h} \leq \text{k}\, Y^{\Delta_i,\nabla_j}_{h-1\leftrightarrow h} = 1, \\ 0, \text{for p} \geq \text{k}\, Y^{\Delta_i,\nabla_j}_{p\leftrightarrow p+1} = 0\,. \end{cases} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots 3$$

The mathematical model is presented in Algorithm 1. Let us suppose a railway line with $\boldsymbol{r}$ stations, $\boldsymbol{n}$ trains running in the down direction, and $\boldsymbol{m}$ trains running in the up direction. We assume that two connected stations have only one line connecting them.

The main complexity of the problem derives in solving the MIP problem due to the decision variables. If we are able to assign values to these decision variables, the linearized problem can be solved more efficiently. Therefore, the main goal of our heuristics is to find values for these decision variables. This assignment will be carried out by means of local search and railway topological knowledge.

The objective function is devoted to travel time. This function reflects mainly the users concerns; however, the railway companies are also interested in saving time due to the more efficient usage of rolling stocks. There are hundreds of studies being undertaken for evaluating savings in travel time (for a review see **[23]**). The passengers want to reach the destination as soon as possible to carry on their activities.

Thus the objective function for this scheduling problem is minimizing the traveling time

$$Min \sum_{i=1}^{n}\left(A_r^{\Delta_i} - D_1^{\Delta_i}\right) + \sum_{j=1}^{m}\left(A_1^{\nabla_j} - D_r^{\nabla_j}\right) \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots .4$$

The objective function minimizes the travel time of all trains in both directions.

Regarding the constraints, there are three groups of scheduling rules in our railway system: traffic rules, user requirements rules, and topological rules. A valid running map must satisfy and optimize the above rules. These scheduling rules can be modeled using the following constraints:

Traffic rules guarantee crossing, expedition, and reception operations. The main constraints are the following.

Regarding the constraints, there are three groups of scheduling rules in our railway system:

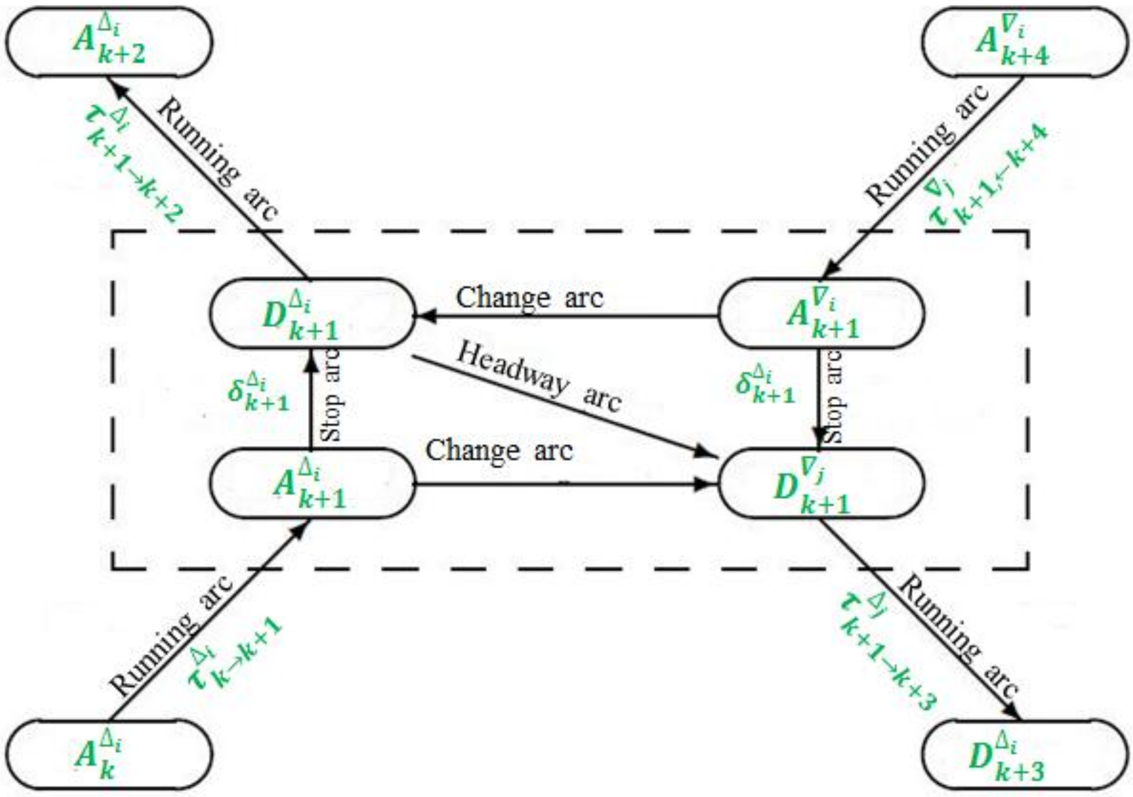
1. *traffic rules,*
2. *user requirements rules, and*
3. *topological rules*
4. *Passenger characteristics*

   *All the mathematical formulations in this section are referred from **[28]** with customized notations.*

A valid *train timetable* must satisfy and optimize the above four rules.

5.2.3 PROBLEM CONSTRAINTS

The previous three scheduling rules can be modeled using the following constraints:



TOPOLOGICAL CONSTRAINTS (TC)

Topological railway infrastructure and type of trains to be scheduled give rise to other constraints to be taken into account. Some of them are what follows. Number of *tracks in station* (to perform technical and/or commercial operations) and the number of tracks between two locations (one-way or two-way) can restrict the operation type of trains movement. No crossing or overtaking is allowed on a one-way track between stations unless there is a siding track.

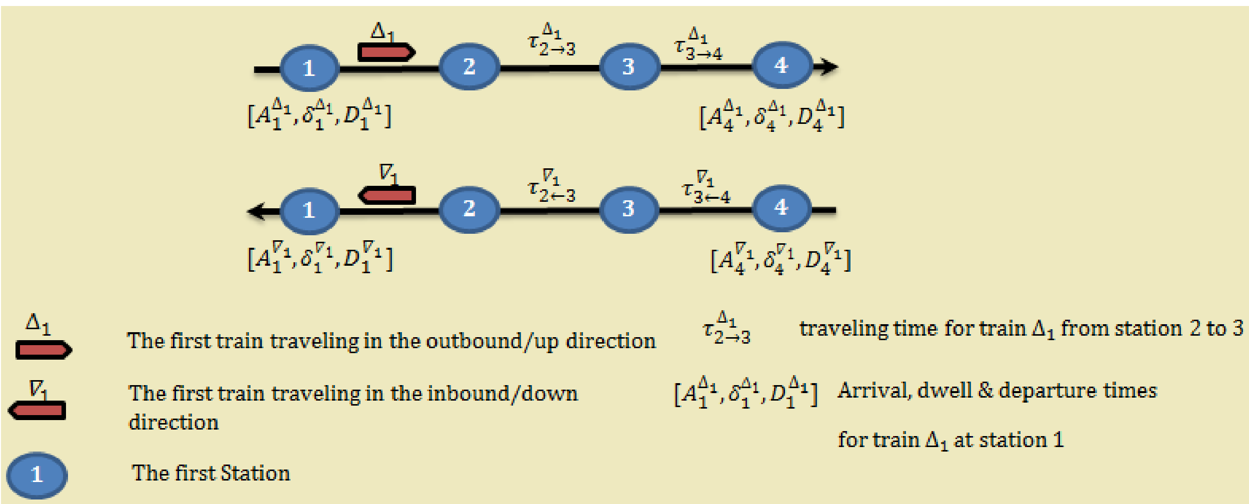


Figure 5.1: TTT variables & rail network topological drawing for double track LRT system

TC.1. *Travel time constraints* determine the necessary time to travel between each two consecutive stations. Thus, constraints (3.1) and (3.2) represent these constraints:

$$Outboundnd\ trains\colon\ A_{k+1,r}^{\Delta_i} - D_k^{\Delta_i} = \tau_{k,r\to k+1,r}^{\Delta_i} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots..\ldots.5a$$

$$Inbound\ trains\colon\ A_{k,r}^{\nabla_j} - D_{k+1,r}^{\nabla_j} = \tau_{k,r\to k+1,r}^{\nabla_j} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots..\ldots.5b$$

TC.2. *Station time/Dwell time constraints* restrict the needed time of each train for technical and/or commercial purposes. It is the time duration that a train stops in a station. This constraint connects arrival and departure event of a train. Dwell times should be long enough for boarding of new passengers and possibly for some loading/unloading or maintenance work on the train. It should not be much longer than necessary, however, as travelers would like to move on and platform capacity within a station might be small.

Thus, constraint (6a) & (6b) determine that commercial stop time is the difference between the departure and the arrival of a train in each station, minus the technical stop time:

$$Outbounnd\ trains\colon A_k^{\Delta_i} - D_k^{\Delta_i} \le TS_{k,r}^{\Delta_i} + CS_{k,r}^{\Delta_i} = \delta_{k,r}^{\Delta_i} \ldots\ldots\ldots\ldots\ldots.\ldots\ldots\ldots\ldots\ldots\ldots.6a$$

$$Inbound\ trains\colon\ A_{k,r}^{\nabla_j} - D_{k,r}^{\nabla_j} \le TS_k^{\nabla_j} + CS_{k,r}^{\nabla_j} = \delta_{k,r}^{\nabla_j} \ldots\ldots\ldots\ldots\ldots\ldots\ldots..\ldots\ldots\ldots.\ldots.6b$$

5.2.4 Dwelling/Commercial stop time estimation

The dwelling time (Commercial stop) model to be estimated is of the following form:

Dwell time at the station with the highest passenger volumes often will control line capacity. Dwell time is frequently the dominant factor in determining the minimum train headway and, thus, the line capacity. The three main components of dwell time are

- Door open and close time, and time waiting to depart once the doors close,
- Passenger flow time, and
- Time the doors remain open after passenger flow ceases.

Of these three factors, passenger flow time is the largest and the hardest to control. It is dependent on passenger volumes at stations, the number of doors on a train, the door widths, the level of crowding inside the train and on the platform, and congestion between boarding and alighting passengers at the train door. The other two factors are, to a great degree, under an agency's control. Minimizing the time spent in a station without passenger flows occurring is important in maintaining reliable train operations, particularly when a line is operating near capacity. The calculation of dwell times is formulated as follows:

$$\delta_{k,r}^{\Delta_i} = T_d = T_o + T_c + A_a N_a + A_b N_b \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots..7$$

Where

T_o	is door open time (2 seconds);
T_c	is door clearance time (8 seconds);
N_a, N_b	are the highest number of alighting and boarding passengers at the door, respectively
A_a, A_b	are the per passenger alighting and boarding time, respectively, and are the parameter to be estimated

Dwelling time estimation at a station

$$\mathbf{T_d = T_o + T_c + A_a N_a + A_b N_b} \quad \textbf{Door system}$$

$$\mathbf{T_d = 3s + 3s + \left[\frac{1s}{\frac{4\ passengers}{door}} \times \frac{317passengers}{4\ doors}\right] \times 2 = 58.83s \approx 1}\text{min}$$

$$\mathrm{T_d} = 3\mathrm{s} + 3\mathrm{s} + \left[\frac{1\mathrm{s}}{\frac{4\ \text{passengers}}{\text{door}}} \times \frac{254\ \text{passengers}}{4\ \text{doors}}\right] \times 2 = 37.75\mathrm{s} \approx 1\mathrm{min},$$

$\boldsymbol{T_o}$	door open time (3± 0.5 seconds) [7] LRT system design
$\boldsymbol{T_c}$	door close time (3± 0.5 seconds)
$\boldsymbol{A_a, A_b}$	per passenger alighting and boarding time, respectively, and are the parameters to be estimated.
$\boldsymbol{N_a, N_b}$	the highest number of alighting and boarding passengers at the door, respectively. Peak hr = 317, off peak hr = 254.

TC.3. *Constraints to limit journey/travel time* restrict allowed time for each train to make the entire travel. Thus, constraints (8a) and (8b) determine that the journey time of each train must be lower than the allowed time plus an allowed margin δ:

$$A_{s,r}^{\Delta_i} - D_{1,r}^{\Delta_i} \le \left(1 + {}^{\delta}/_{100}\right) \times \tau_{1,r\to s,r}^{\Delta_i} \quad \dots \dots 8a$$

$$A_{1,r}^{\nabla_j} - D_{s,r}^{\nabla_j} \le \left(1 + {}^{\delta}/_{100}\right) \times \tau_{1,r\to s,r}^{\nabla_j} \quad \dots \dots 8b$$

TC.4. In accordance with user requirements, the system should obtain the best solution available so that all the above constraints are satisfied. Several criteria can exist to qualify the optimality of solutions: minimize duration and/or number of technical stops, minimize the total time of train trips (span) of the total schedule, giving priority to certain trains, and so on.

Trip time: The trip time is the time needed for the train to run between two stations. Trip times do not necessarily need to be fixed, but can also be variable, as reported in [7]. The lower bound for the trip time is the minimum time needed for the train to run the distance plus a reserve of a few percent that helps making the schedule more robust. The upper bound is the maximum acceptable time with respect to passenger patience and track capacity usage. The trip time (l, u) is a constraint between the departure and arrival events of the same train.

Headway: The headway constraints are used to avoid collisions. They separate two trains running on the same track by at least the headway time h. This is done by introducing constraints (h; T - h) between the arrival and the departure events of the two trains. It guarantees that the departures and arrivals of the two trains on the same track have a safe temporal distance. The headway time is only a simplification of the real safety system used in the railway world. More precise safety restrictions should be taken into account during the micro scheduling.

TRAFFIC RULE CONSTRAINTS (TRC)

Traffic rules guarantee crossing and overtaking operations. We assume two trains ($i \ and \ j$) going in opposite directions between stations $k \ and \ k + 1$. The main constraints to take into account are:

TRC_① *Crossing Constraints: 9a and 9b*

Any two trains going in opposite directions must not simultaneously use the same one-way track:

$$A_{k,r}^{\nabla_j} - D_{k,r}^{\Delta_i} \leq UB \times Y_{k,r \rightarrow k+1,r}^{\Delta_i,\nabla_j} \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots 9a$$

$$A_{k+1,r}^{\Delta_i} - D_{k+1,r}^{\nabla_j} \leq UB \times (1 - Y_{k,r \rightarrow k+1,r}^{\Delta_i,\nabla_j}) \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots 9b$$

We can use the following constraints in place of the above inequalities.

$$(D_k^{\Delta_i} + \tau_{k\_k+1}^{\Delta_i} < D_{k+1}^{\nabla_j}) \wedge (D_k^{\Delta_i} < D_{k+1}^{\nabla_j} + \tau_{k+1\_k}^{\nabla_j}) \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots .9a'$$

$$\vee$$

$$(D_k^{\Delta_i} + t_{k\_k+1}^{\Delta_i} > D_{k+1}^{\nabla_j}) \wedge (D_k^{\Delta_i} > D_{k+1}^{\nabla_j} + t_{k+1_k}^{\nabla_j}) \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots 9b'$$

The crossing of two trains can be performed only on two-way tracks and at stations, where one of the two trains has been detoured from the main track. Constraints (9a or 9a') and (9b or 9b') that represent the *disjunctive constraint* which restricts two trains going in opposite directions require the same section of track at the same time.

TRC_② *Overtaking Constraint*

Two trains ($\Delta_i \ and \ \Delta_s$) going at different speeds in the same direction can only overtake each other at stations.

$$(D_k^{\Delta_i} < D_k^{\Delta_s}) \wedge (D_k^{\Delta_i} + t_{k\_k+1}^{\Delta_i} < D_k^{\Delta_s} + t_{k\_k+1}^{\Delta_s}) \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots 10a$$

$$\vee$$

$$(D_k^{\Delta_i} > D_k^{\Delta_s}) \wedge (D_k^{\Delta_i} + t_{k\_k+1}^{\Delta_i} > D_k^{\Delta_s} + t_{k\_k+1}^{\Delta_s}) \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots 10b$$

TRC_③ *Expedition Time Constraint*

There exist a given time to put a train back on the main track so that crossing or overtaking can be performed (ET). Or on the other hand at least are required ET^{Δ_i} time units at location $\boldsymbol{k}$ between the arrival and departure times of two trains ∇_j and ∇_i going in the opposite direction. Thus, constraints (11a) and (11b) that represent the disjunctive constraint which restricts two trains going in opposite directions require a *buffer time* between the arrival and departure times of two trains:

$$A_{k,r}^{\nabla_j} - D_{k,r}^{\Delta_i} - UB \times \left(X^{\Delta_i,\nabla_j} - Y_{k,r\leftrightarrow k+1,r}^{\Delta_i,\nabla_j} + Y_{k+1,r\leftrightarrow k+2,r}^{\Delta_i,\nabla_j} - 1\right) + ET^{\Delta_i} \leq 0 \ldots\ldots\ldots\ldots\ldots\ldots\ldots .11a$$

$$A_{k,r}^{\Delta_i} - D_{k,r}^{\nabla_j} - UB \times \left(X^{\Delta_i,\nabla_j} - Y_{k,r\leftrightarrow k+1,r}^{\Delta_i,\nabla_j} + Y_{k+1,r\leftrightarrow k+2,r}^{\Delta_i,\nabla_j} - 2\right) + ET^{\nabla_j} \leq 0 \ldots\ldots\ldots\ldots\ldots\ldots\ldots .11b$$

Or you can use the following logic equivalent to the above inequality

$$(D_k^{\Delta_i} + ET_i < D_k^{\nabla_j}) \vee (D_k^{\nabla_j} + ET^{\nabla_j} < D_k^{\Delta_i})$$

TRC_④ Reception Time Constraint

At least are required RT_i time units at location k between the arrival times of two trains T_j and T_i going in the opposite direction. Thus, constraints (12a) and (12b) that represent the disjunctive constraint which restricts two trains going in opposite directions require a buffer time between the arrivals of two trains:

$$A_{k,r}^{\Delta_i} - A_{k,r}^{\nabla_j} - UB \times \left(X^{\Delta_i,\nabla_j} - Y_{k,r\leftrightarrow k+1,r}^{\Delta_i,\nabla_j} + Y_{k+1,r\leftrightarrow k+2,r}^{\Delta_i,\nabla_j} - 1\right) + RT^{\Delta_i} \leq 0 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots .12a$$

$$A_{k,r}^{\nabla_j} - A_{k,r}^{\Delta_i} - UB \times \left(X^{\Delta_i,\nabla_j} - Y_{k,r\leftrightarrow k+1,r}^{\Delta_i,\nabla_j} + Y_{k+1,r\leftrightarrow k+2,r}^{\Delta_i,\nabla_j} - 2\right) + RT^{\nabla_j} \leq 0 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots .12b$$

On the other hand the above constrain can be performed in a different way, i.e. there exist a given time to detour a train back from the main track so that crossing or overtaking can be performed (RT).

$$(A_k^{\Delta_i} + RT^{\Delta_i} < A_k^{\nabla_j}) \vee (A_k^{\nabla_j} + RT^{\nabla_j} < D_k^{\Delta_i})$$

In this research work the four traffic rule constraints discussed above won't be considered in the solution part. Because they are applied for bidirectional operation of trains along a single track which may occur in Addis Ababa LRT system only in case of track segment unavailability by maintenance of the track segment or deadlock situations. It will be good to solve the problem including these constraints if the time available to wind up the research is not too short. Hence it is better leave it for now and recommend those who are interested to work their research on train schedule to include the bidirectional operational conditions i.e. when trains are allowed to move in opposite direction on a single track. The assumptions made to solve the problem for this paper in section………

USER REQUIREMENT CONSTRAINT (URC)

The main constraints due to user requirements are:

URC.1. T*ype and number of trains* going in each direction to be scheduled,

URC.2. *Path of trains:* Locations used and *stop time* for commercial purposes in each direction.

i. $D_k^{\Delta_i} = A_k^{\Delta_i} + ST_k^{\Delta_i}$

URC.3. *Scheduling Frequency/headway constraint*: The frequency constraint specifies the period (Freq) between departures of two consecutive trains in each direction at the same location. This constraint is very restrictive because, when crossing is performed, trains must wait for a certain time interval at stations. This interval must be propagated to all trains going in the same direction in order to maintain the established scheduling frequency. The user can require a fixed frequency, a frequency within a minimum and maximum interval, or multiple frequencies. Thus, constraint (2) restricts that the time period between departures of two consecutive trains at location *k* must be equal to Freq:

$$D_{k,r}^{\Delta_{i+1}} - D_{k,r}^{\Delta_i} = headway = H = Freq \text{ ……………………………………………………} 13a$$

$$D_{k,r}^{\nabla_{j+1}} - D_{k,r}^{\nabla_j} = headway = H = Freq \text{ ……………………………………………………} 13b$$

On the other hand train departure must satisfy frequency requirements in both directions. It could be fixed time (a) or a time interval (Freq $\pm\, \delta$) (b). Frequency is a very tight constraint is only sometimes required.

$$A_k^{\Delta_{i+1}} = D_k^{\Delta_i} + Freq \text{ ……………………………………………………} 14a$$

$$D_k^{\Delta_i} + Freq - \delta \le D_k^{\Delta_{i+1}} \le (D_k^{\Delta_i} + Freq + \delta) \text{ ………………………………} 14b$$

URC.4. **Departure interval:** Departure interval for the departure of the first trains going in both the up and down directions.

i. $StartTime^{\Delta_i} < D_1^{\Delta_i} < EndTime^{\Delta_i}$ ………………………………………………15

URC.5. Connection time/Passenger transfer time constraint

These constraints relate the arrival event of some train to the departure event of another one in order to enable passengers to change trains in another route. The minimum connection time depends on the infrastructure of the railway station, on the distances passengers have to walk. Upper bounds are again the acceptable waiting times for the travelers.

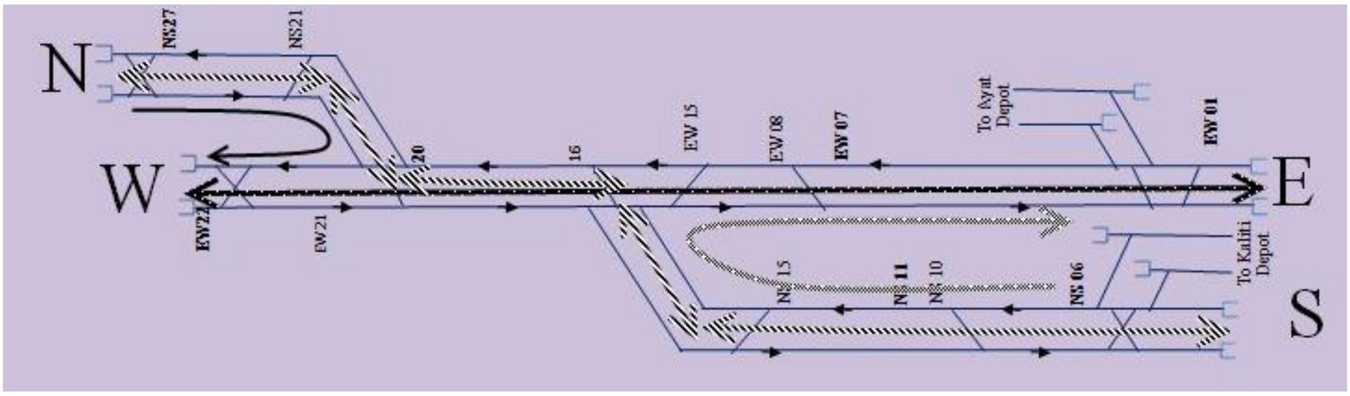


Figure 5.2: Passenger Route changing locations

Key:

passenger transfer path from North end to West end

passenger transfer path from East end to South end

East-West Main route

North-South Main route

Note: For the connection/passenger transfer constraint use the following hints to avoid mismatching of connection

$\tau_{kaliti\_Stadium}$

$\tau_{torhayloch-Lideta}$

$\tau_{Hayat-stadium}$

$\tau_{Piasa-Lideta}$

For passengers who want to move from Hayat to Kaliti or vice versa

We have the following connection time constraints

$\boldsymbol{\tau^{\Delta}_{H\to S}}$	*Travel time from Hayat to stadium in the up direction*
$\boldsymbol{\tau^{\nabla}_{S\to K}}$	*Travel time from stadium to Kaliti in the down direction*
$\boldsymbol{\tau^{\Delta}_{K\to S}}$	*Travel time from Kaliti to stadium in the up direction*
$\boldsymbol{\tau^{\nabla}_{S\to H}}$	*Travel time from stadium to Hayat in the down direction*
$\boldsymbol{t^{c}_{K\to H}}$	*Connection time from Hayat to Kaliti*
$\boldsymbol{t^{c}_{H\to K}}$	*Connection time from Hayat to Kaliti*

Transfer constraint for movement from Hayat to Kaliti

$$\tau^{\nabla}_{S\to K} = \tau^{\Delta}_{H\to S} + t^{c}_{H\to K} \ldots \ldots \ldots 16a$$

Transfer constraint for movement from Kaliti to Hayat

$$\tau^{\nabla}_{S\to H} = \tau^{\Delta}_{K\to S} + t^{c}_{K\to H} \ldots \ldots \ldots 16b$$

……………………………………………………………………………………….

$\boldsymbol{\tau^{\Delta}_{L\to T}}$	*Travel time from Lideta to Torhayloch in the up direction*
$\boldsymbol{\tau^{\nabla}_{P\to L}}$	*Travel time from Piasa to Lideta in the down direction*
$\boldsymbol{\tau^{\Delta}_{L\to P}}$	*Travel time from Lideta to Piasa in the up direction*
$\boldsymbol{\tau^{\nabla}_{T\to L}}$	*Travel time from Torhayloch to Lideta in the down direction*
$\boldsymbol{t^{c}_{P\to T}}$	*Connection time from Piasa to Torhayloch*
$\boldsymbol{t^{c}_{T\to P}}$	*Connection time from Torhayloch to Piasa*

Transfer constraint for movement from Piasa to Torhayloch

$$\tau^{\Delta}_{L\to T} = \tau^{\nabla}_{P\to L} + t^{c}_{P\to T} \ldots \ldots \ldots 17a$$

Transfer constraint for movement from Torhayloch to Piasa

$$\tau^{\Delta}_{L\to P} = \tau^{\nabla}_{T\to L} + t^{c}_{T\to P} \ldots \ldots \ldots 17b$$

SUMMARY OF THE MATHEMATICAL MODEL

PROPOSED MATHEMATICAL MODEL

(4) $\boldsymbol{Min \sum_{i=1}^{n}\left(A_s^{\Delta_i} - D_1^{\Delta_i}\right) + \sum_{j=1}^{m}\left(A_1^{\nabla_j} - D_s^{\nabla_j}\right)}$, Travel time minimization

Subjected to constraints listed from (2) to (9)

Frequency/Headway time constraint: $\forall\, i = 1, \ldots n, \forall\, j = 1, \ldots m,$

$(\boldsymbol{13a})\quad D_{k,r}^{\Delta_{i+1}} - D_{k,r}^{\Delta_i} = headway = H = Freq$ $\quad \forall k = 1, \ldots. s(\#\, of\ stations)$

$(\boldsymbol{13b})\quad D_{k,r}^{\nabla_{j+1}} - D_{k,r}^{\nabla_j} = headway = H = Freq$ $\quad \forall\, r = \{NS\ Line, EW\ Line\}$

Running/traveling time constraints $\forall\, i = 1, \ldots n, \forall\, j = 1, \ldots m,$

$(\boldsymbol{5a}) Outband\ trains\text{:}\ A_{k+1,r}^{\Delta_i} - D_{k,r}^{\Delta_i} = \tau_{k,r\to k+1,r}^{\Delta_i}$ $\quad \forall\, k = 1, \ldots. s$

$(\boldsymbol{5b})\ Inband\ trains\text{:}\ A_{k,r}^{\nabla_j} - D_{k+1,r}^{\nabla_j} = \tau_{k,r\to k+1,r}^{\nabla_j}$ $\quad \forall\, r = \{NS\ Line, EW\ Line\}$

Station dwell/stop time constraint:

$(\boldsymbol{6a})\quad Outbound\text{:}\ A_{k,r}^{\Delta_i} - D_{k,r}^{\Delta_i} \le TS_{k,r}^{\Delta_i} + CS_{k,r}^{\Delta_i} = \delta_{k,r}^{\Delta_i}$ $\quad \forall\, i = 1, \ldots n, \forall\, k = 1, \ldots. r$

$(\boldsymbol{6b})\ Inbound\text{:}\ A_{k,r}^{\nabla_j} - D_{k,r}^{\nabla_j} \le TS_{k}^{\nabla_j} + CS_{k,r}^{\nabla_j} = \delta_{k,r}^{\nabla_j}$ $\quad \forall\, j = 1, \ldots m,$

$\forall\, r = \{NS\ Line, EW\ Line\}$

Constraints to limit total journey time

$$(\mathbf{8a}) \quad A_{s,r}^{\Delta_i} - D_{1,r}^{\Delta_i} \leq \left(1 + {}^{\delta}/_{100}\right) \times \tau_{1,r\to s,r}^{\Delta_i}$$

$$(\mathbf{8b}) \quad A_{1,r}^{\nabla_j} - D_{s,r}^{\nabla_j} \leq \left(1 + {}^{\delta}/_{100}\right) \times \tau_{1,r\to s,r}^{\nabla_j}$$

$\forall\, i = 1, \ldots n, \forall\, k = 1, \ldots .m$:
$\forall\, r = \{NS\ Line, EW\ Line\}$

Crossing time constraints:

$$(\mathbf{9a}) \quad A_{k,r}^{\nabla_j} - D_{k,r}^{\Delta_i} \leq UB \times Y_{k,r\to k+1,r}^{\Delta_i,\nabla_j}$$

$$(\mathbf{9b}) \quad A_{k+1,r}^{\Delta_i} - D_{k+1,r}^{\nabla_j} \leq UB \times (1 - Y_{k,r\to k+1,r}^{\Delta_i,\nabla_j}).$$

$\forall\, i = 1, \ldots n, \forall\, j = 1, \ldots .m, \quad \forall\, k = 1, \ldots s$:
$\forall\, r = \{NS\ Line, EW\ Line\}$

Expedition time constraints:

$$(\mathbf{11a}) \quad A_{k,r}^{\nabla_j} - D_{k,r}^{\Delta_i} - UB \times \left(X^{\Delta_i\nabla_j} - Y_{k,r\leftrightarrow k+1,r}^{\Delta_i\nabla_j} + Y_{k+1,r\leftrightarrow k+2,r}^{\Delta_i\nabla_j} - 1\right) + ET^{\Delta_i} \leq 0$$

$$(\mathbf{11b}) \quad A_{k,r}^{\Delta_i} - D_{k,r}^{\nabla_j} - UB \times \left(X^{\Delta_i\nabla_j} - Y_{k,r\leftrightarrow k+1,r}^{\Delta_i\nabla_j} + Y_{k+1,r\leftrightarrow k+2,r}^{\Delta_i\nabla_j} - 2\right) + ET^{\nabla_j} \leq 0$$

$\forall\, i = 1, \ldots n, \forall\, j = 1, \ldots .m, \quad \forall\, k = 1, \ldots r$

Reception time constraints

$$(\mathbf{12a}) \quad A_{k,r}^{\Delta_i} - A_{k,r}^{\nabla_j} - UB \times \left(X^{\Delta_i\nabla_j} - Y_{k,r\leftrightarrow k+1,r}^{\Delta_i\nabla_j} + Y_{k+1,r\leftrightarrow k+2,r}^{\Delta_i\nabla_j} - 1\right) + RT^{\Delta_i} \leq 0$$

$$(\mathbf{12b}) \quad A_{k,r}^{\nabla_j} - A_{k,r}^{\Delta_i} - UB \times \left(X^{\Delta_i\nabla_j} - Y_{k,r\leftrightarrow k+1,r}^{\Delta_i\nabla_j} + Y_{k+1,r\leftrightarrow k+2,r}^{\Delta_i\nabla_j} - 2\right) + RT^{\nabla_j} \leq 0$$

$\forall\, i = 1, \ldots n, \forall\, j = 1, \ldots .m, \quad \forall\, k = 1, \ldots s$:

Binary constraints :

$$(2) \quad X^{\Delta_i\nabla_j} = \begin{cases} 0, \text{if train } \Delta_i \text{ arrives at the crossing station first} \\ 1, \text{ if train } \nabla_j \text{ arrives at the crossing station first} \end{cases}$$

$\forall\, i = 1, \ldots n, \forall\, j = 1, \ldots .m$

$$(3) \quad Y_{k,r\leftrightarrow k+1,r}^{\Delta_i\nabla_j} = \begin{cases} 1, \ \text{for h} \leq \text{k}\ Y_{h-1,r\leftrightarrow h,r}^{\Delta_i\nabla_j} = 1 \\ 0, \text{for p} \geq \text{k}\ Y_{p,r\leftrightarrow p+1,r}^{\Delta_i\nabla_j} = 0 \end{cases}$$

$\forall\, i = 1, \ldots n, \forall\, j = 1, \ldots .m, \forall\, k = 1, \ldots .s$

Meeting or Transfer time constraints

$(16a)\tau_{S\to K}^{\nabla} = \tau_{H\to S}^{\Delta} + t_{H\to K}^{c}$, for stadium meeting point
$(16b)\tau_{S\to H}^{\nabla} = \tau_{K\to S}^{\Delta} + t_{K\to H}^{c}$, for stadium meeting point
$(17a)\tau_{S\to K}^{\nabla} = \tau_{H\to S}^{\Delta} + t_{H\to K}^{c}$, for Lideta meeting point
$(17b)\tau_{L\to P}^{\Delta} = \tau_{T\to L}^{\nabla} + t_{T\to P}^{c}$, for Lideta meeting point

Considered path for transfer

Hayat to Kaliti
Kaliti to Hayat
Piasa to Torhayloch
Torhayloch to Piasa

Table 5.1: LRT train schedule mathematical model

NOTATIONS USED IN THE MODEL

Notations	
s	Number of stations
n	Number of trains running in the down direction
m	Number of trains running in the up direction
H	Headway time between departures of two consecutive trains

$A_{k,r}^{\Delta_i}$	Arrival time of train Δ_i at station k which is found on route r
$D_{k,r}^{\Delta_i}$	Departure time of train Δ_i from station k which is located on route r
$\tau_{k,r\to k+1,r}^{\Delta_i}$	Traveling/running time of train Δ_i from station k to k+1
$\tau_{1,r\to r,r}^{\Delta_i}$	Total traveling/running time of train Δ_i to travel from station 1 to final station s on the route r
δ	Allowed percentage of a train to arrive at destination
$\delta_{k,r}^{\Delta_i}$	Dwell Time at station k which is located on the route r
$TS_{k,r}^{\Delta_i}$	Technical stop time of train Δ_i at station k
$CS_{k,r}^{\Delta_i}$	The commercial stop time of train Δ_i at station k which is located on route r
ET^{Δ_i}	The expedition time of train Δ_i
RT^{Δ_i}	The reception time of train Δ_i
UB	The upper bound of the train travel for this line
$Y_{k,r\to k+1,r}^{\Delta_i,\nabla_j}$	This variable determines the track between station k and station $k+1$ in which train Δ_i crosses with train ∇_j
X^{Δ_i,∇_j}	This variable determines which train (Δ_i *or* ∇_j) arrives earlier to the crossing station. If $X^{\Delta_i,\nabla_j} = 0$, then train Δ_i arrives at the crossing station first and train ∇_j arrives at the same station later
Δ_i	Train i moving in the up or outbound direction
∇_j	Train j moving in the down or inbound direction
$\tau_{L\to T}^{\Delta}$	Travel time from Lideta to Torhayloch in the up direction
$\tau_{P\to L}^{\nabla}$	Travel time from Piasa to Lideta in the down direction
$\tau_{L\to P}^{\Delta}$	Travel time from Lideta to Piasa in the up direction
$\tau_{T\to L}^{\nabla}$	Travel time from Torhayloch to Lideta in the down direction
$t_{P\to T}^{c}$	Connection time from Piasa to Torhayloch
$t_{T\to P}^{c}$	Connection time from Torhayloch to Piasa
$\tau_{H\to S}^{\Delta}$	Travel time from Hayat to stadium in the up direction
$\tau_{S\to K}^{\nabla}$	Travel time from stadium to Kaliti in the down direction
$\tau_{K\to S}^{\Delta}$	Travel time from Kaliti to stadium in the up direction
$\tau_{S\to H}^{\nabla}$	Travel time from stadium to Hayat in the down direction
$t_{K\to H}^{c}$	Connection time from Hayat to Kaliti
$t_{H\to K}^{c}$	Connection time from Hayat to Kaliti

Table 5.2: Notation used for LRT train schedule model

CHAPTER 6

6.1. SCHEDULING ALGORITHM

The scheduling algorithm used for this research work is a sequential algorithm since the operation of AA LRT trains is of such type. The complete algorithm is shown below.

AA LRT System Train Schedule Algorithm

```
begin  Sequential algorithm
   getInputForSched (initialDep, firstStation, lastStation, route, train)
       station_ew = fisrtStation_ew
       station_ns = fisrtStation_ns
       DepTime=initialDep_ew=initialDep_ns=initialDep
      While (station_ew != last Station_ew || station_ns != last Station_ns )
             i= getTrains_ew (train_ew)
             j= getTrains_ns (train_ns)
             if(track !=sharedLine)
                foreach (train_i  in train_ew || train_j  in train_ns )
                          if (train_i = first train_ew && train_j = first train_ns)
                               setArrivTime_firstTrain_ew_firstStation = operationStartTime
                                setArrivTime_firstTrain_ns_firstStation = operationStartTime
                                depTime_firstTrain_ew = operationStartTime + dwellTime_AtfirstStation
                                depTime_firstTrain_ns = operationStartTime + dwellTime_AtfirstStation
                                makeSchedule(train_i,station_j)
                          end if
                          else
                            nextStation_ew = getNextStation_ew (station_ew)
                            nextStation_ns = getNextStation_ns (station_ns)
                            train_i_ArrivTime_Sation_j=train_i_DepTime_Sation_(j-1) + runningTime
                            train_i_DepTime_Sation_j = train_i_ArrivTime_Sation_j + dwellTime_j
                            makeSchedule(train_i,station_j)
                          end else
                     VerifySafetyHeadwayBetweenSuccessiveTrains(train_i,train_i+1)
                      makeSchedule(train_i,station_j)
             end if
        else
            verify_MeetingConstraints(MeetingStation, nextStation , depTime, train_i)
            makeSchedule(train_i,station_j)
            station = nextStation
            train=nextTrain
        end else
     end while
end sequential algorithm
```

6.2 INTERFACING SQL SERVER DATABASE WITH MS VISUAL STUDIO.NET IDE

For this research work the LRT timetable/database is created using Microsoft SQL Server 2008 relational database management system (RDMS) which is integrated in Visual Studio.NET integrative software development environment. The timetable displays train arrival time, departure time, train ID, train route and direction. The timetable can be updated using ADO.NET programming with Visual C Sharp, see figure 5.1 how this framework operates to communicate with the database. ADO.NET is a data access API for the .NET Framework. It provides a set of class libraries used to access and update the data in a database which can be used to develop database driven applications with c# as well as the other .NET languages such as VB.NET and ASP.NET. Due to its popularity and my familiarity with visual c#.NET programming language I preferred it for this research work.

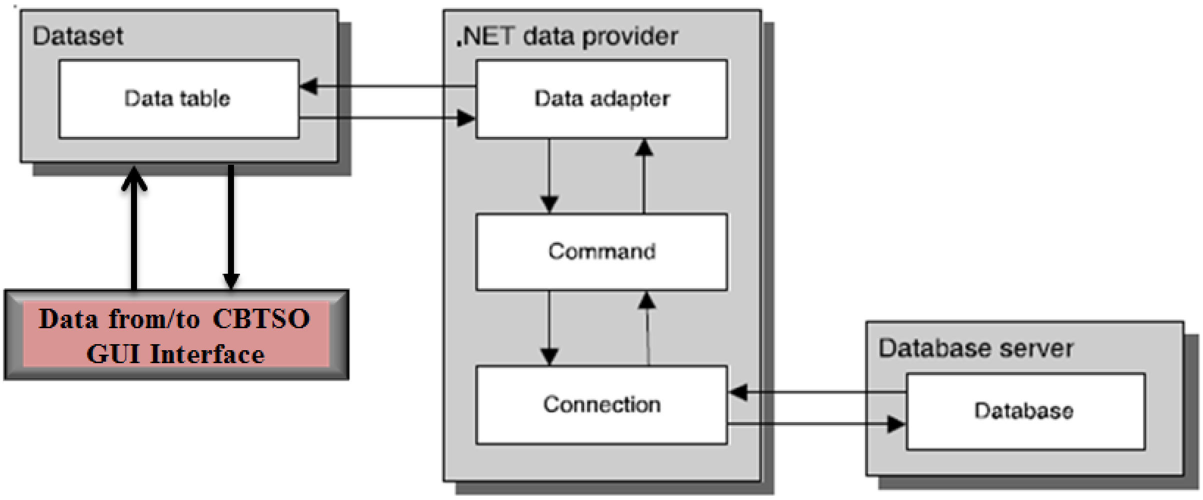


Figure 6.1. BASIC ADO.NET object component closed loop diagram

Microsoft SQL Server 2008 is used here to create the LRT database used for timetable generation. The database contains a set of tables. There are totally six tables namely: Train table, station table, operation day table, Track segment table, timetable table and route table. In order to integrate the database with CBTSO GUI interface Visual C sharp programming language and SQL statements are embedded into the C# code behind the GUI application. Programming development environment used for this research is called Visual Studio.NET, see figure 5.4. The SQL server database is also developed using this programming tool using its built in MS SQL server 2008.

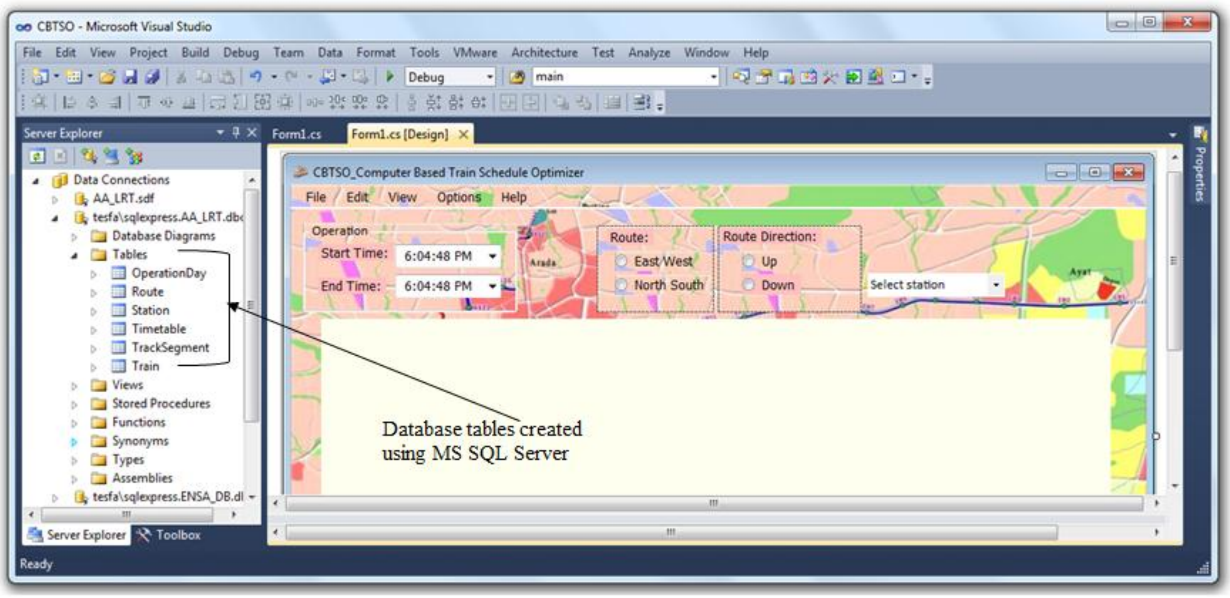


Figure 6.2 Microsoft SQL server database developed using Microsoft Visual Studio IDE

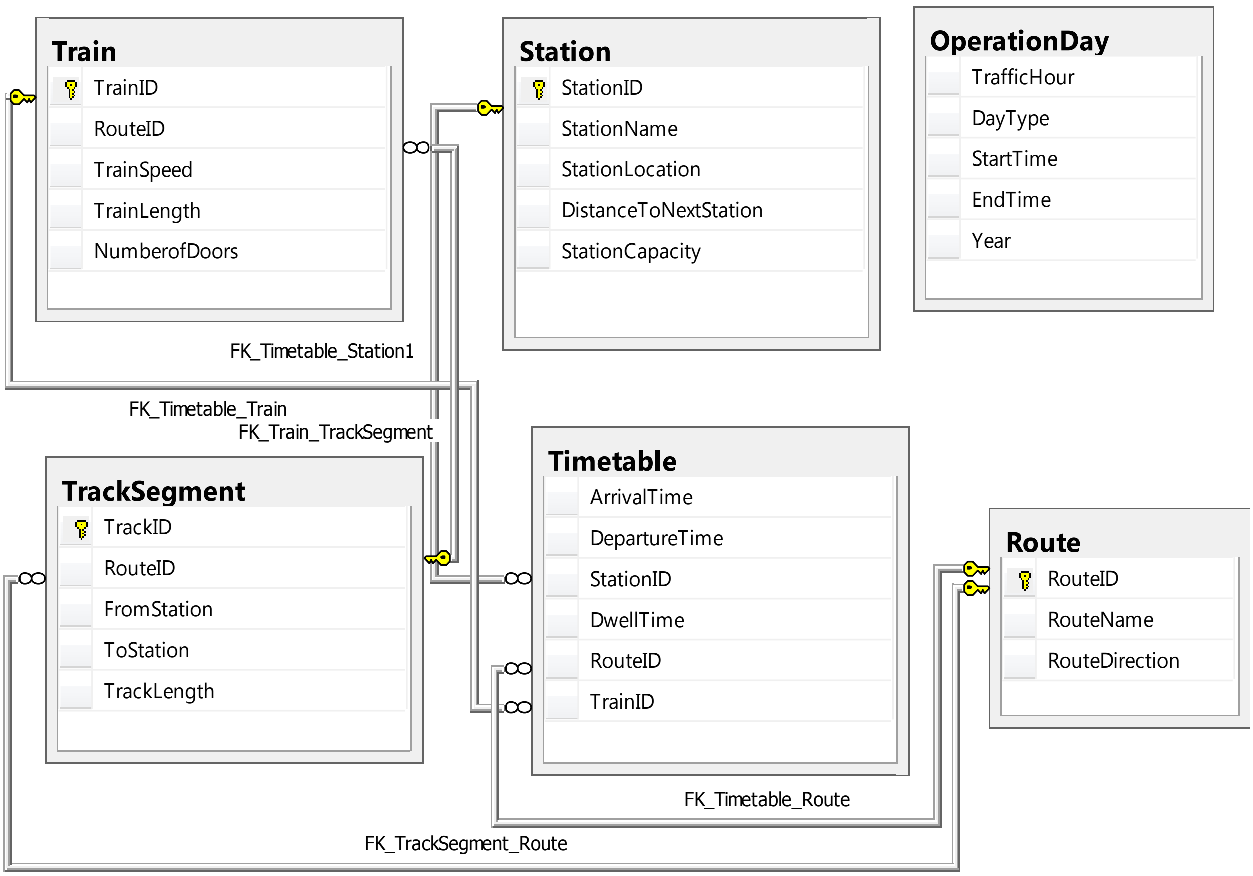


Figure 6.3: CBTSO database entity diagram

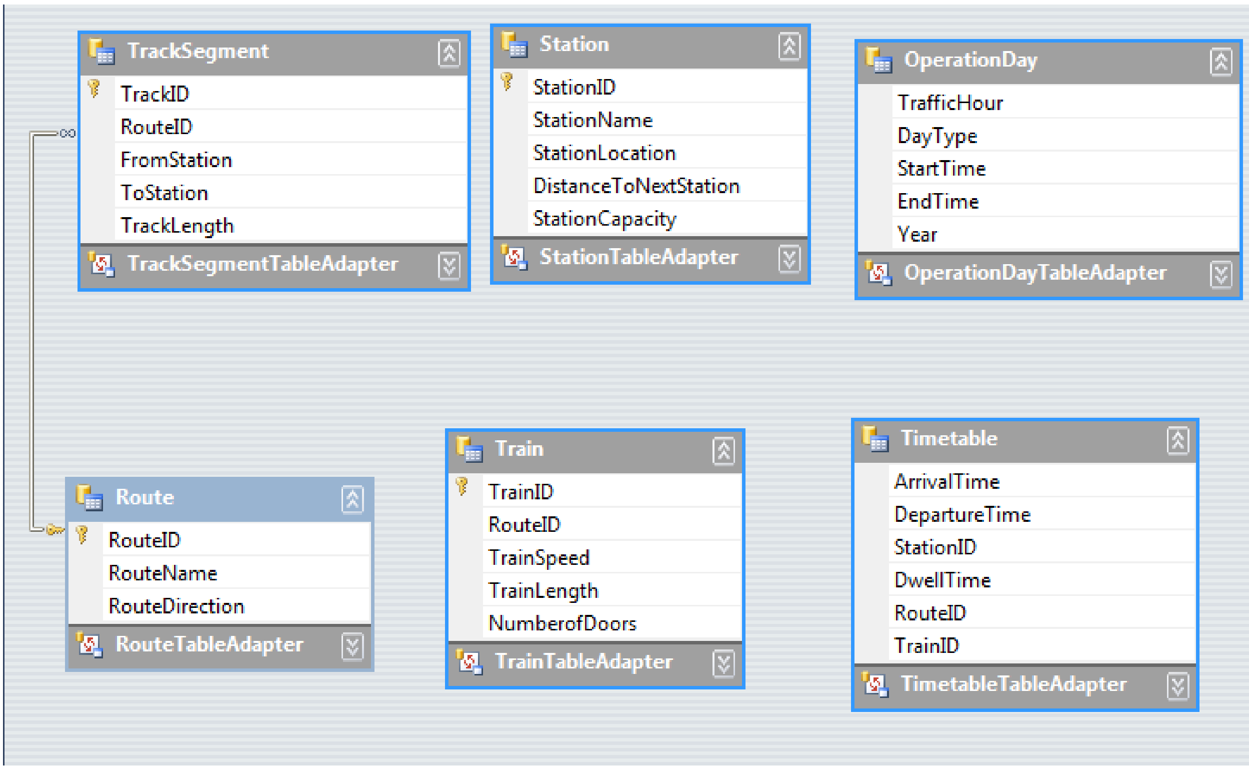


Figure 6.2 Microsoft Visual Studio 2010 IDE

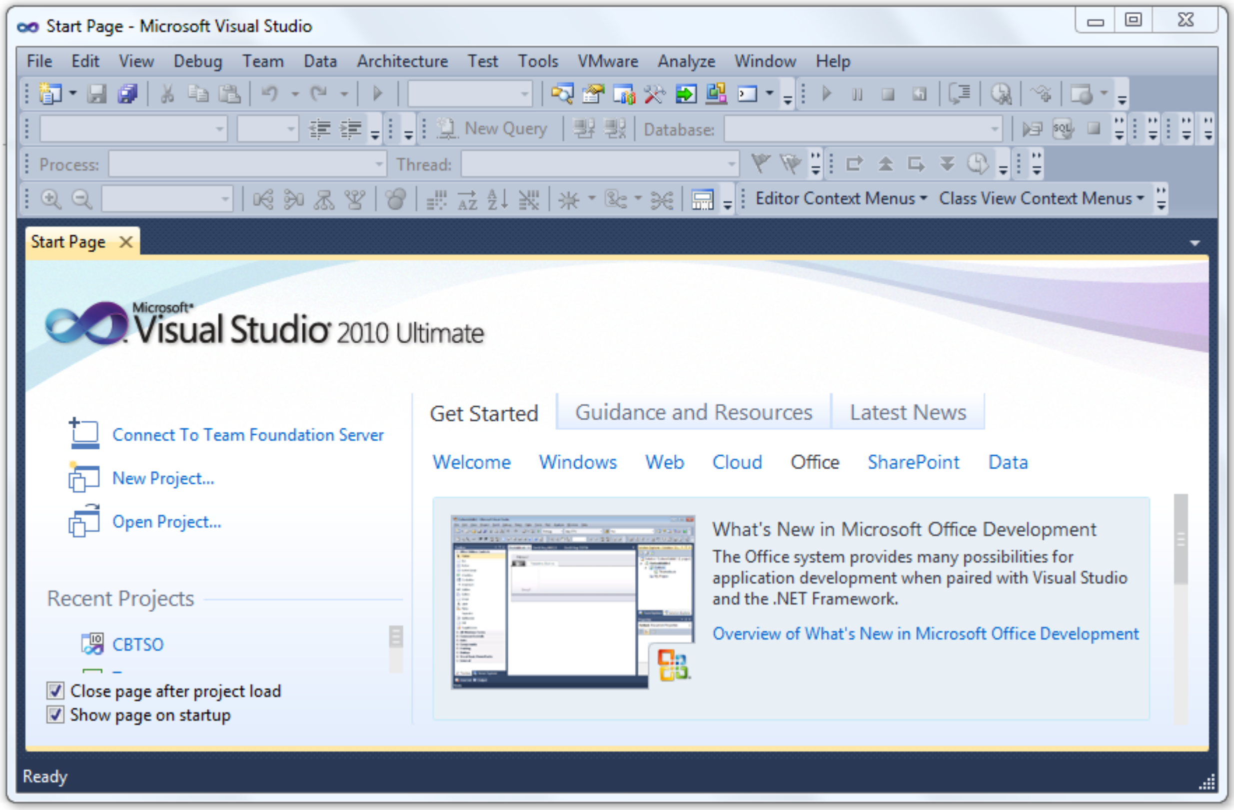


Figure 6.3 Microsoft Visual Studio 2010 IDE

For this research the ultimate version of Microsoft Visual Studio 2010 is used in order to generate the entire GUI interface of CBTSO using visual C#.NET 4.0 programming language. The database used for CNTSO scheduling software in for this research work is composed of seven tables as it shown in figure 6.6.

Three different programs used to generate train schedule for AA-LRT are presented in appendix B.1. These visual C sharp dot net programs were made in such a manner that they were interfaced with Ms-SQL server database using ADO.NET database linking programming technology.

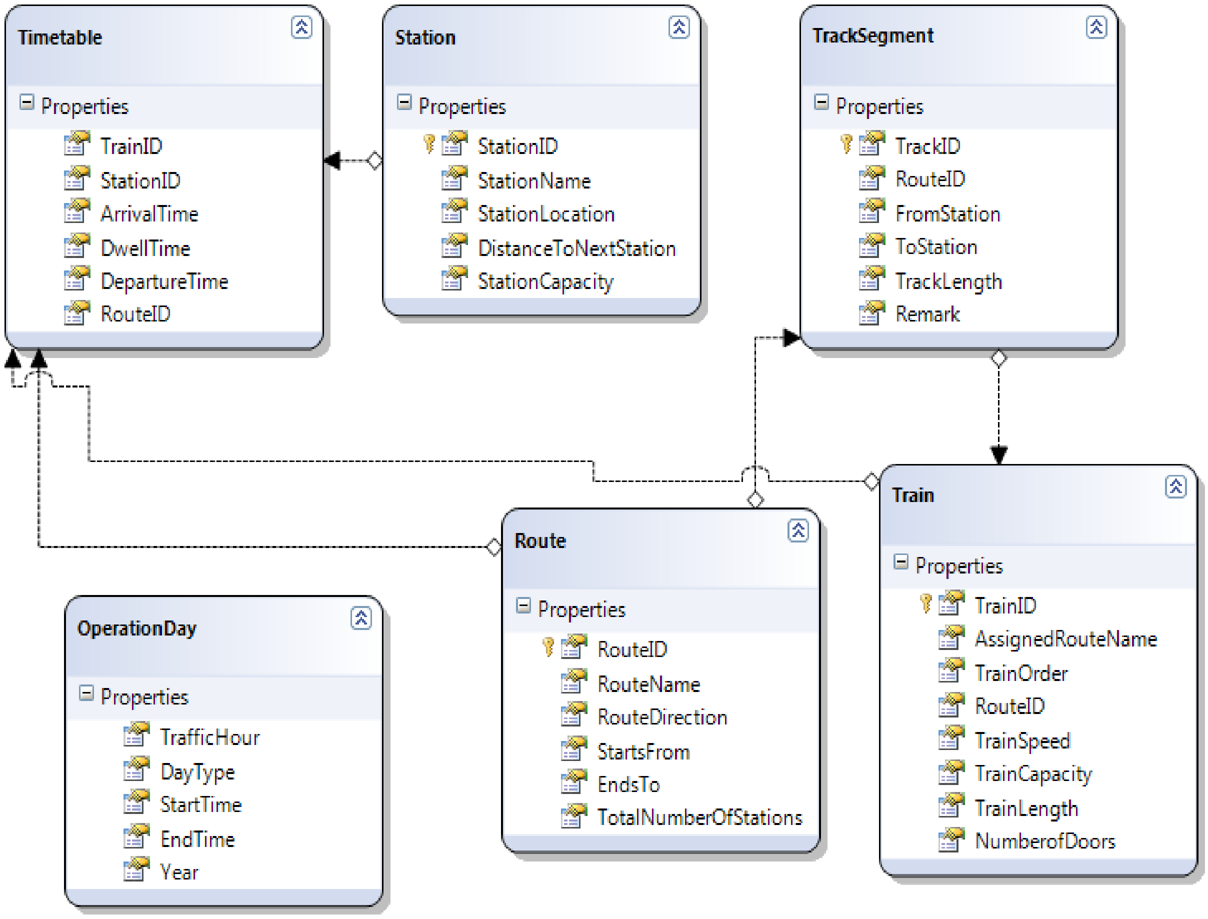


Figure 6.4: Entity data model diagram of CBTSO

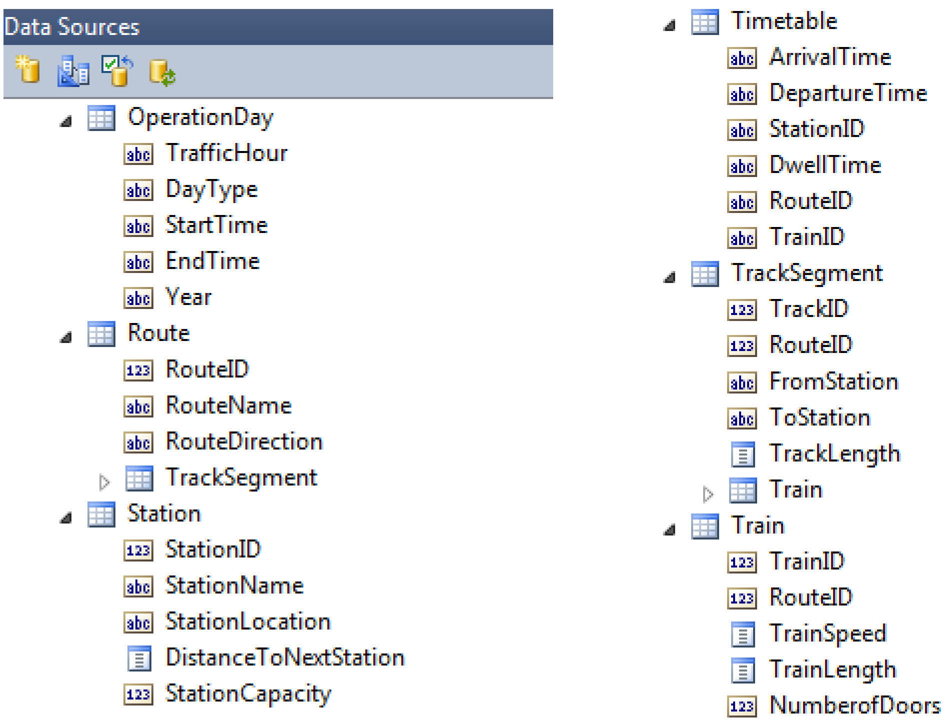


Figure 6.5: Data Source that are fetched into the dataset container

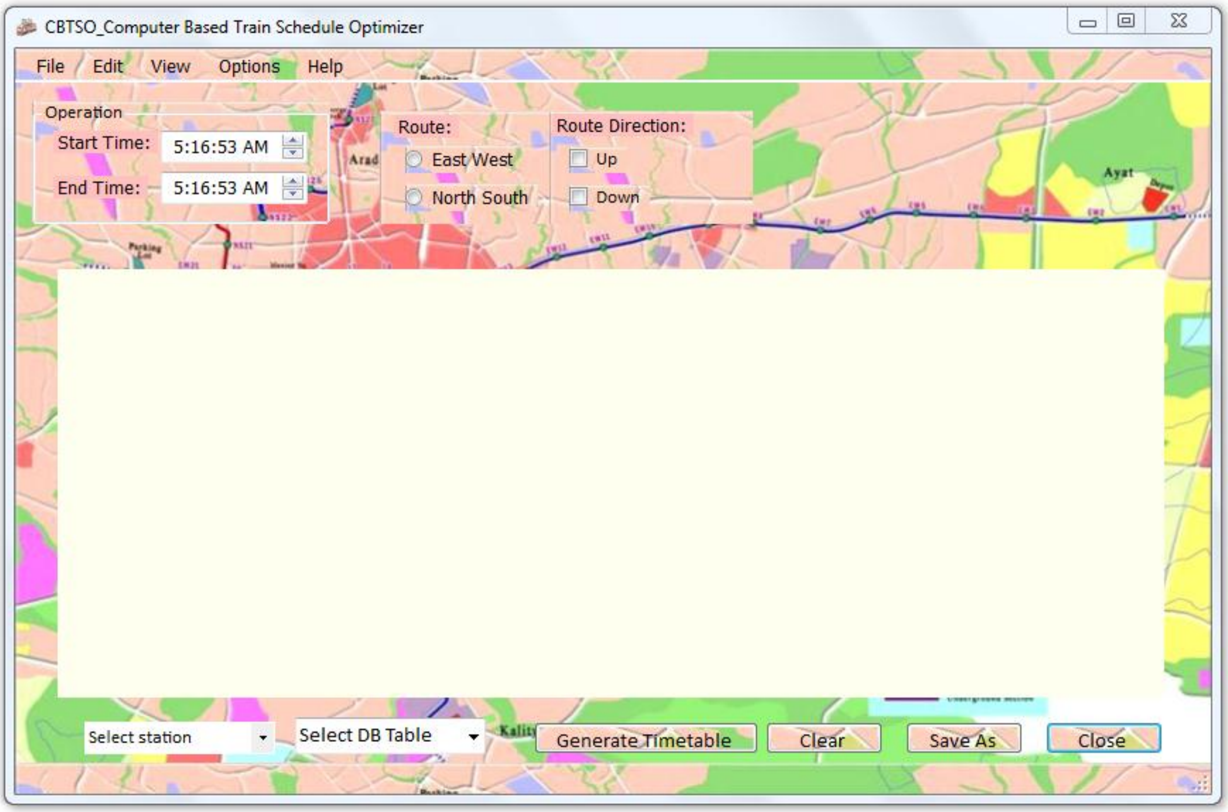


Figure 6.6: CBTSO application snapshot at the first startup

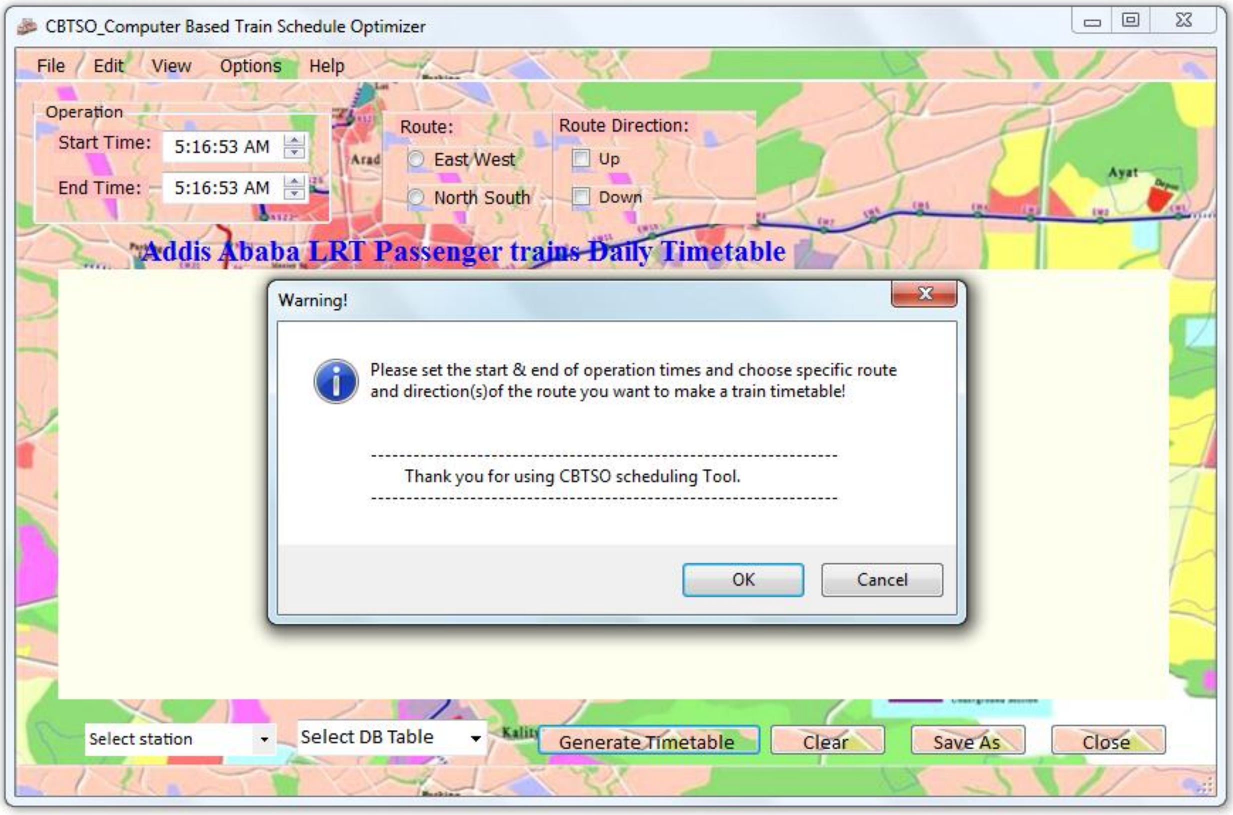


Figure 6.7. A snap shot when generate timetable button is clicked without choosing a specific train route and route direction

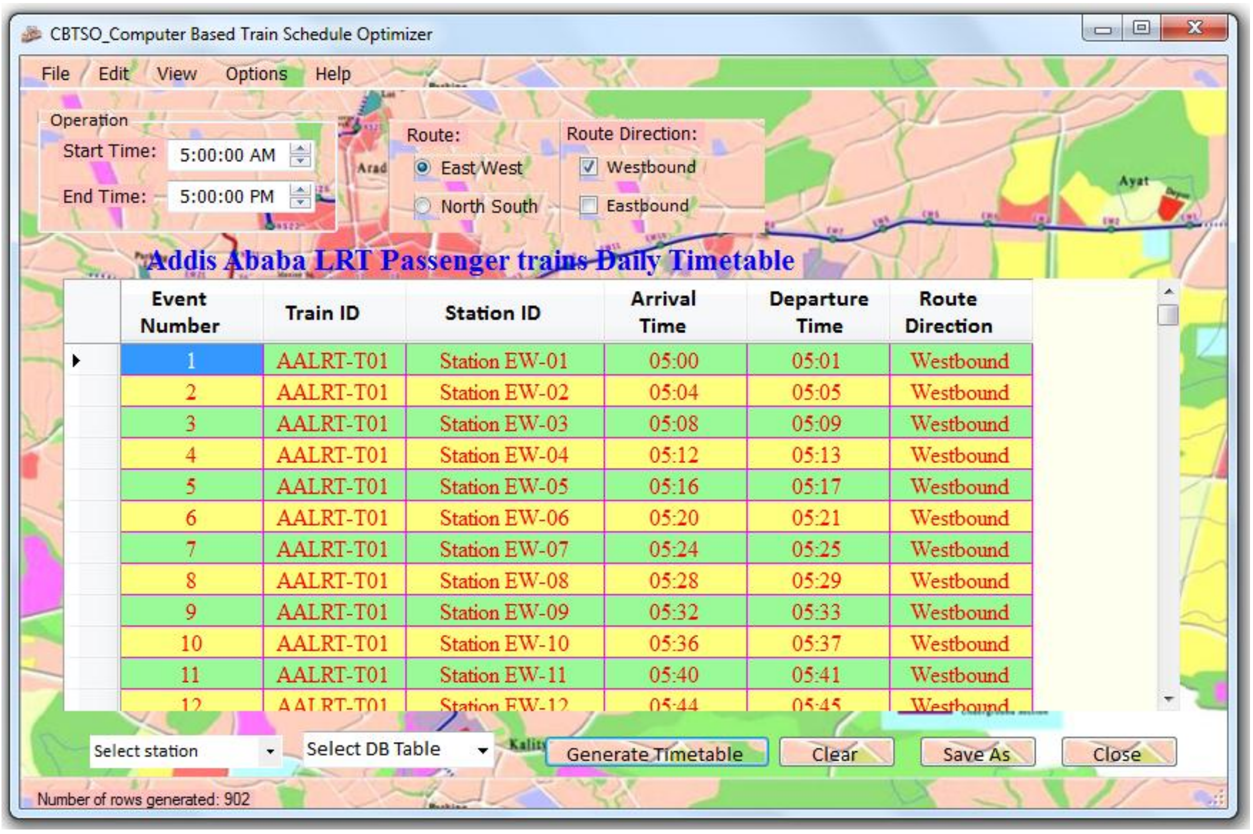


Figure 6.8:Result of timetable for East-West route in the westbound direction

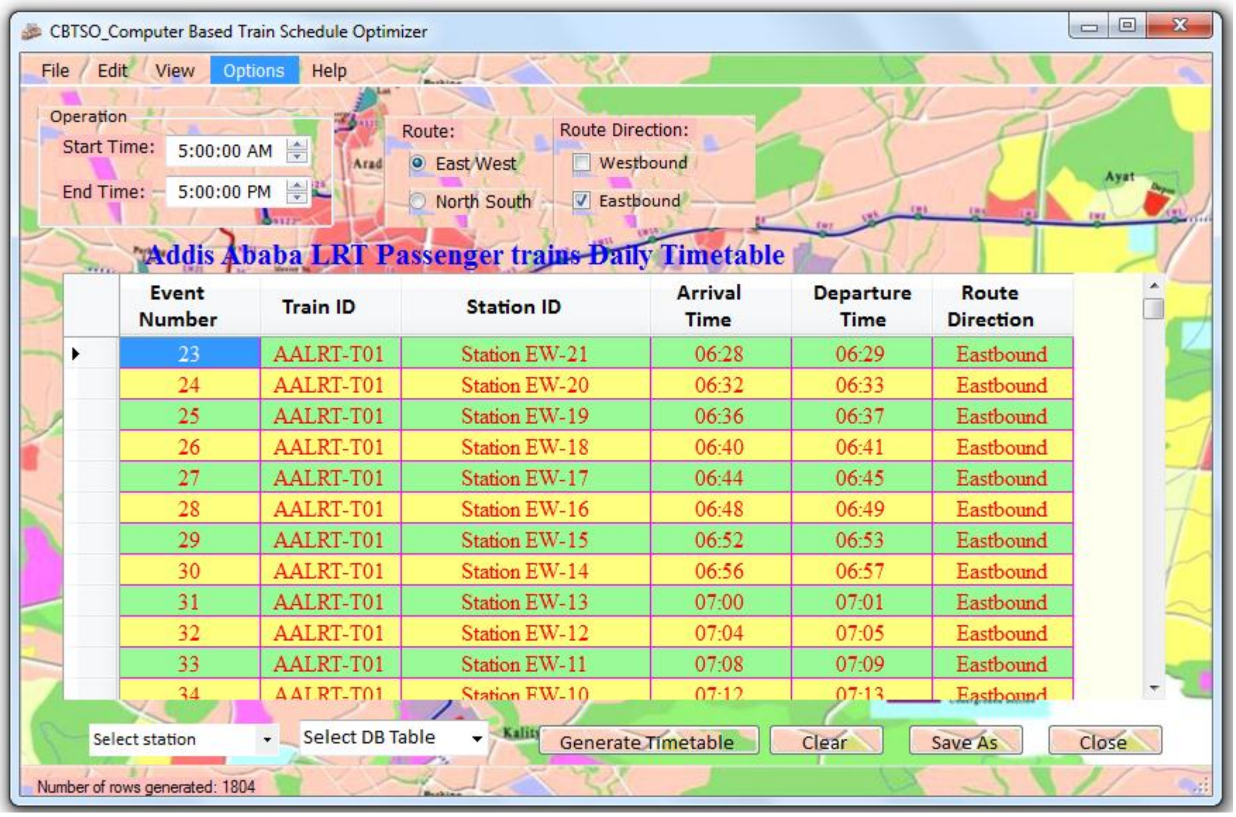


Figure 6.9. Result of timetable for East-West route in the eastbound direction

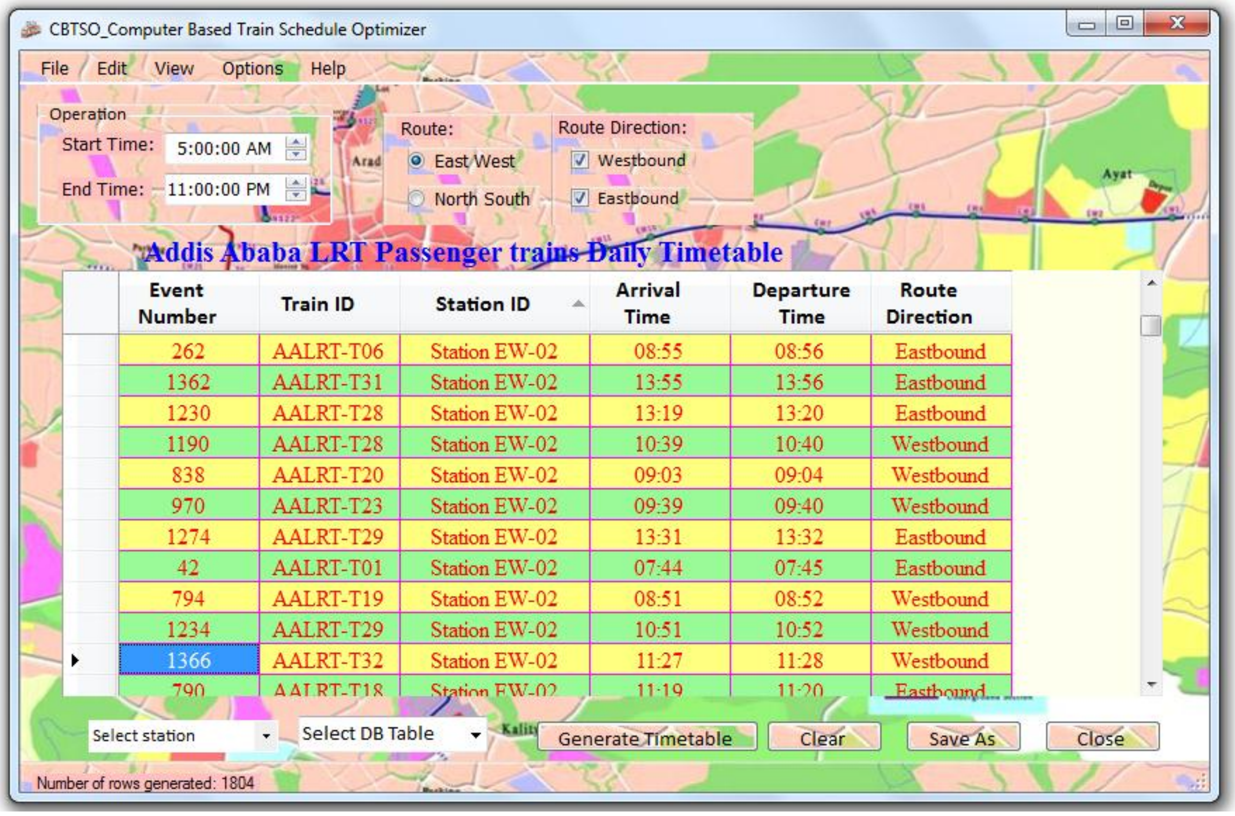


Figure 6.10 Result of timetable for East-West route in both the westbound and eastbound directions sorted by station ID

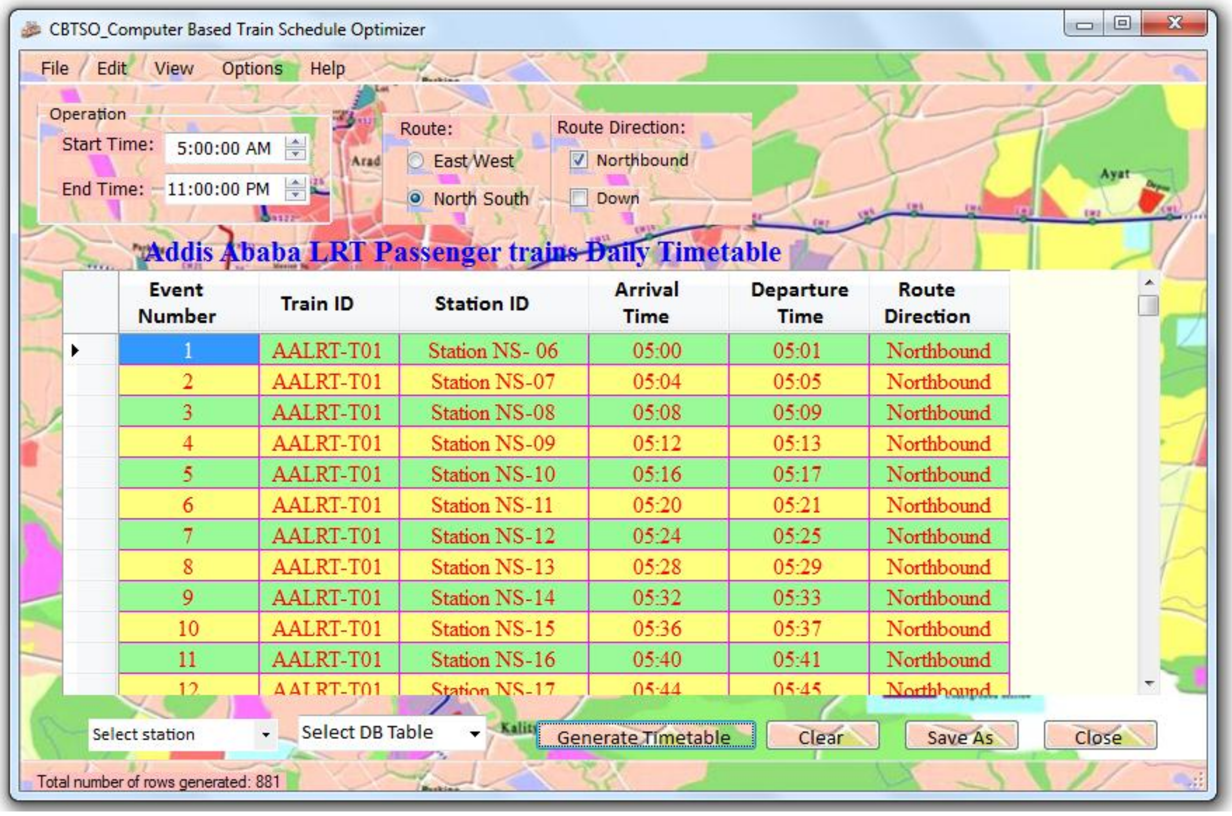


FIGURE 6.11. Result of timetable for North-South route in both the northbound directions

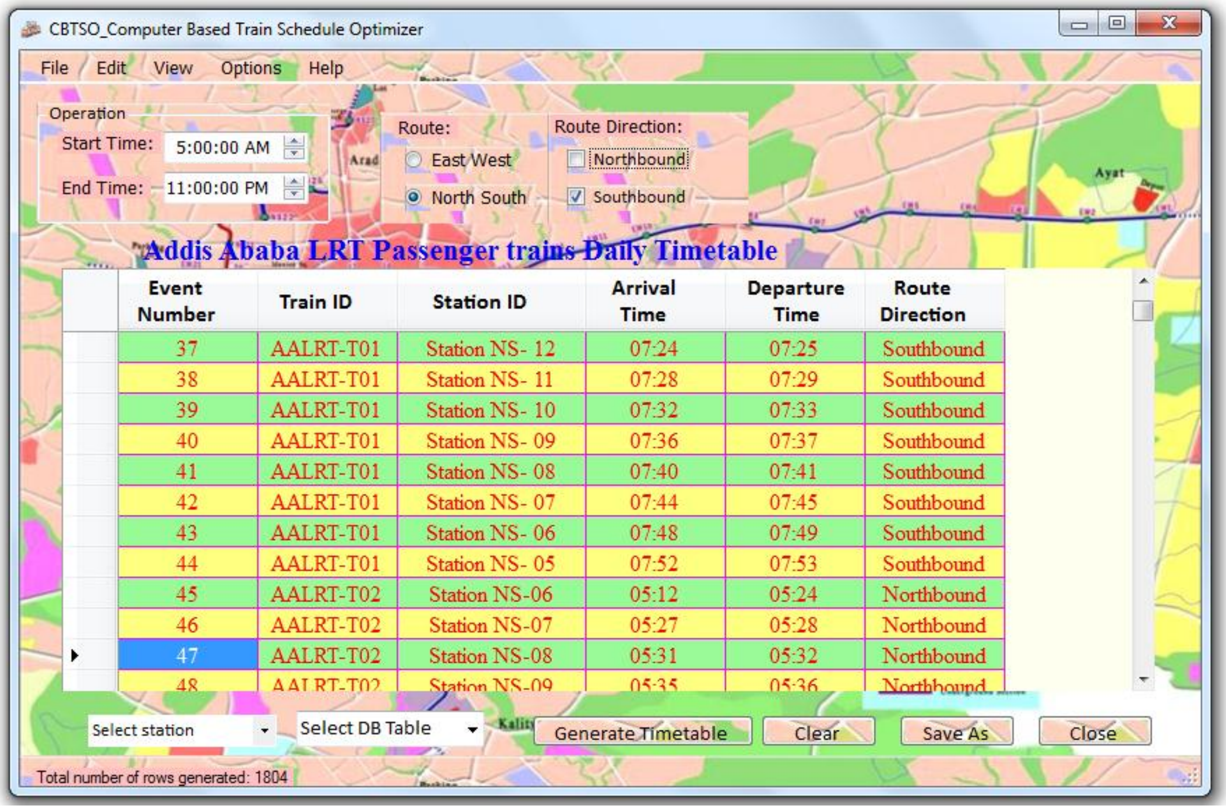


Figure 6.12 Result of timetable for North-South route in the southbound direction

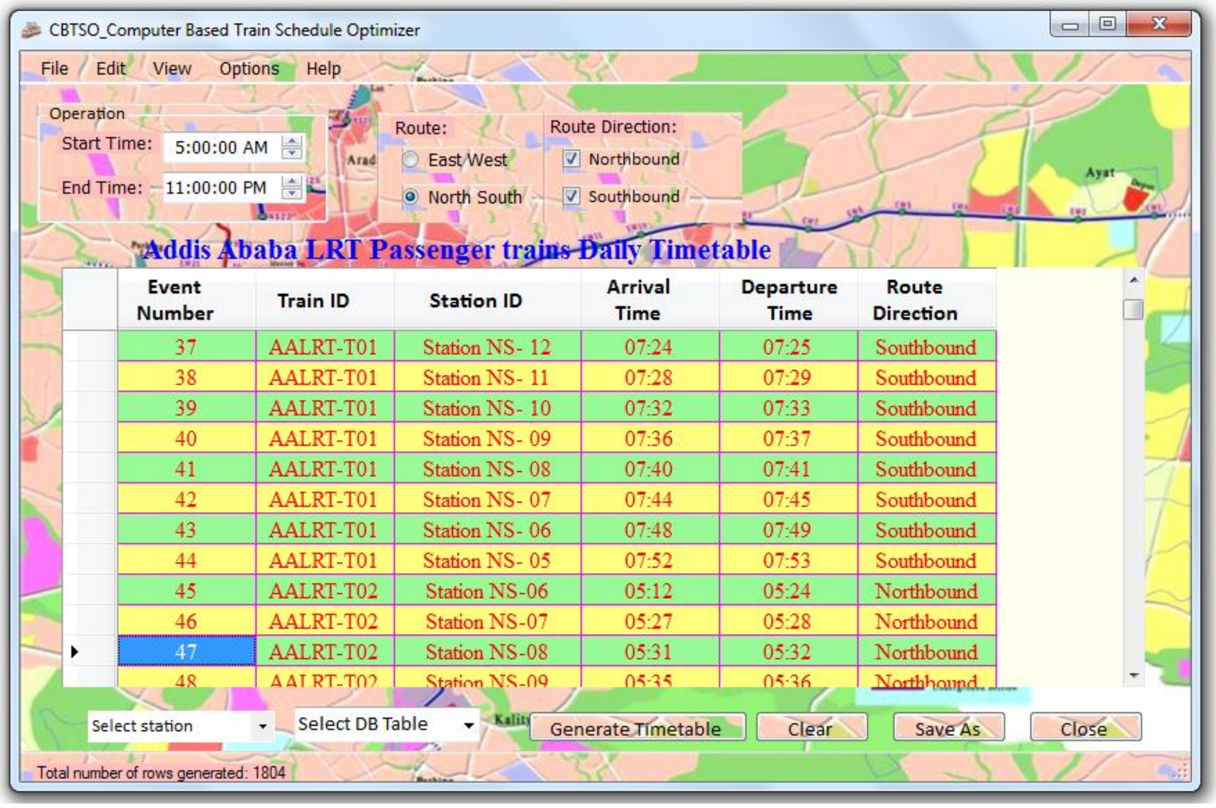


Figure 6.13. Timetable for North-South route in both northbound and southbound directions

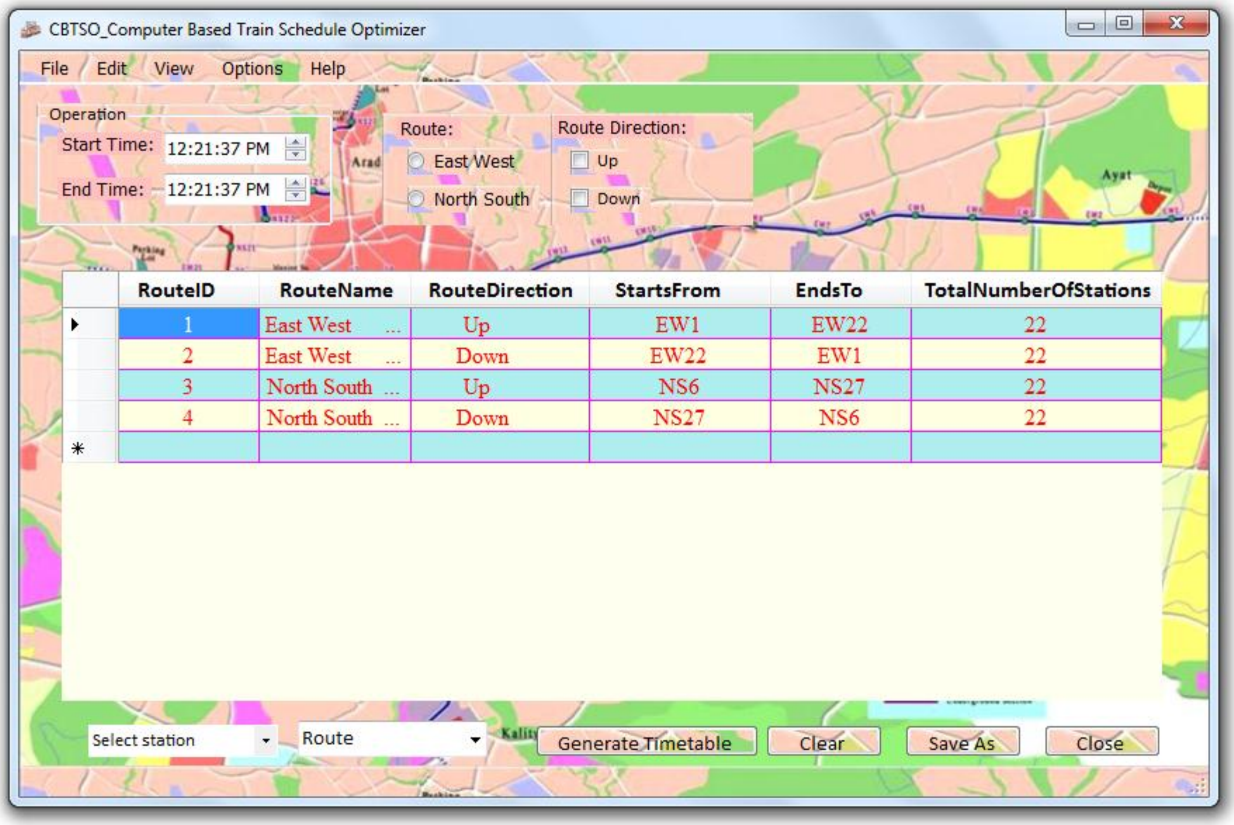


Figure 6.14: LRT route details

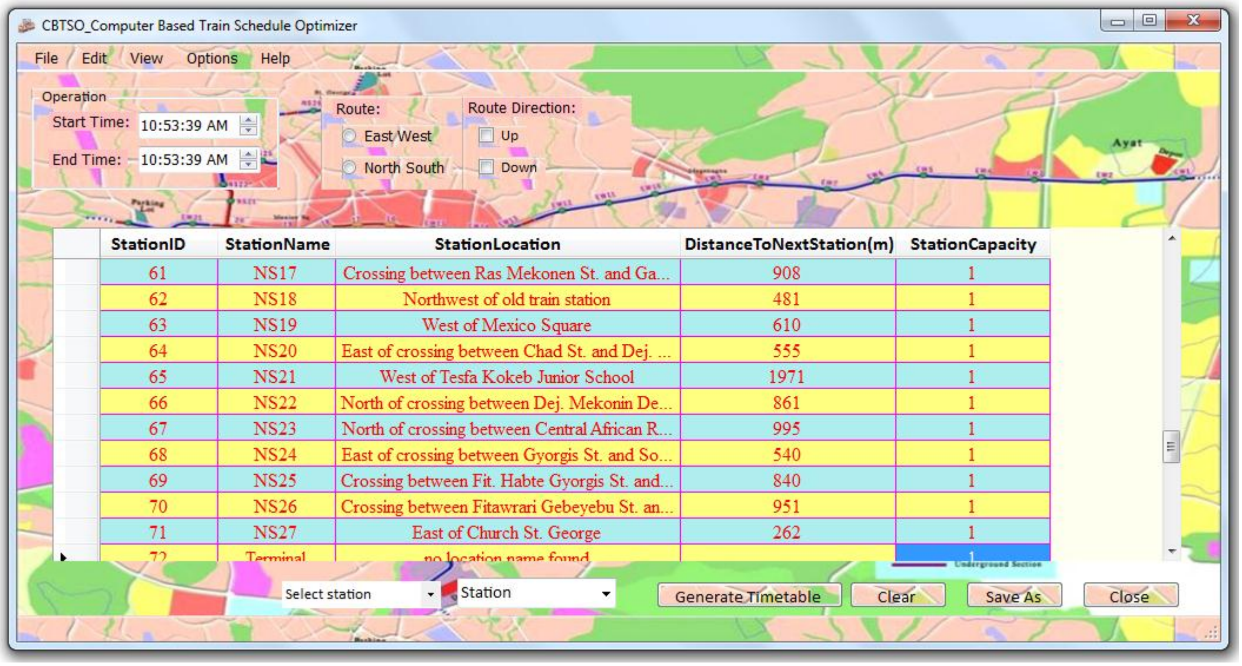


Figure 6.15 LRT station details

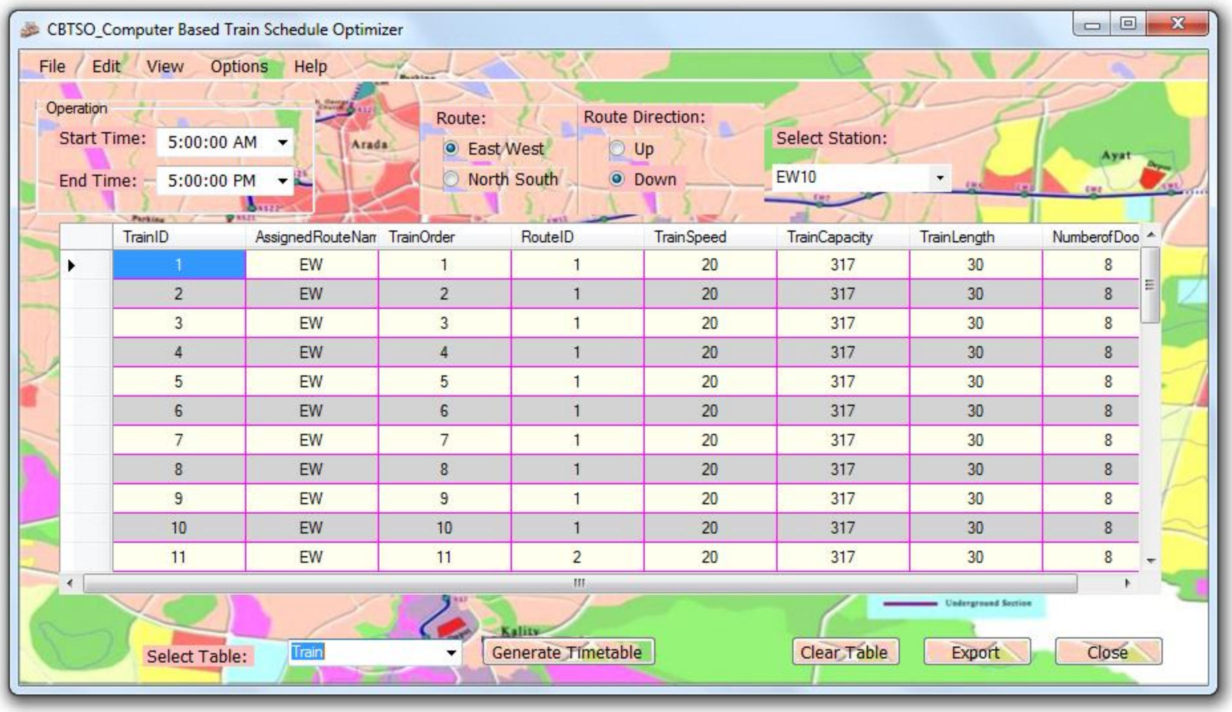
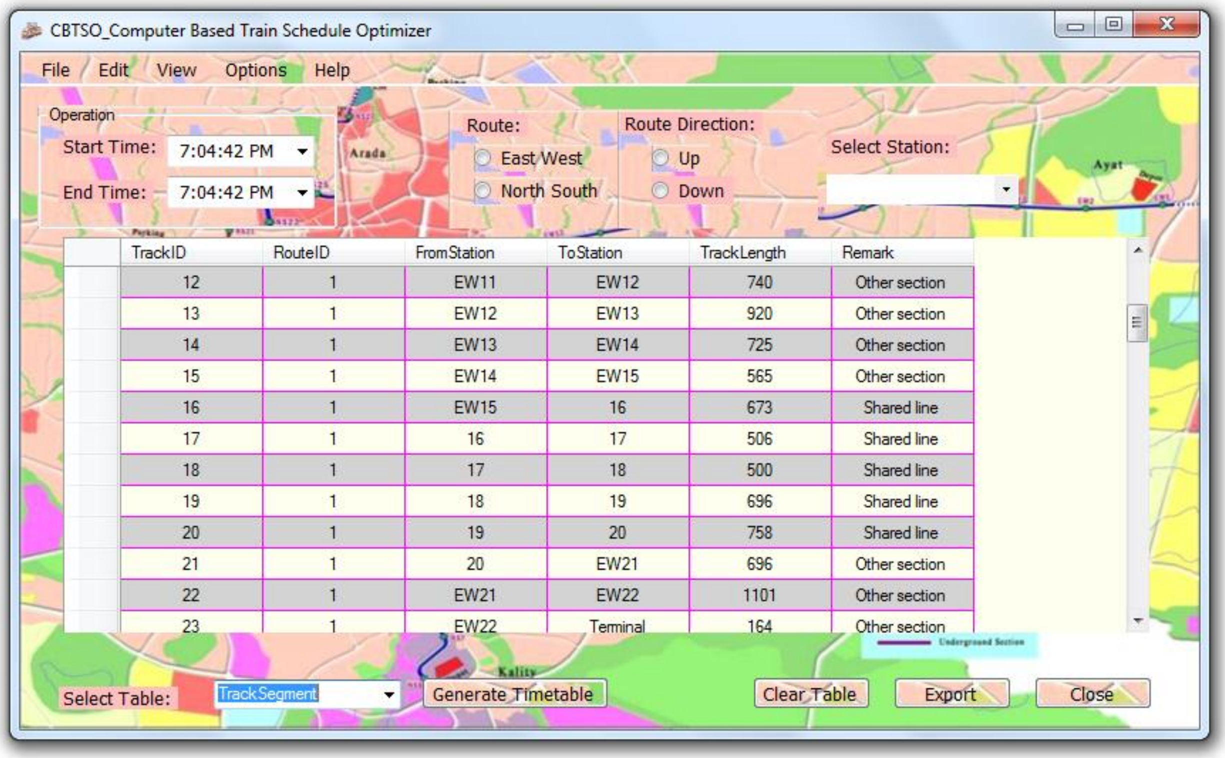
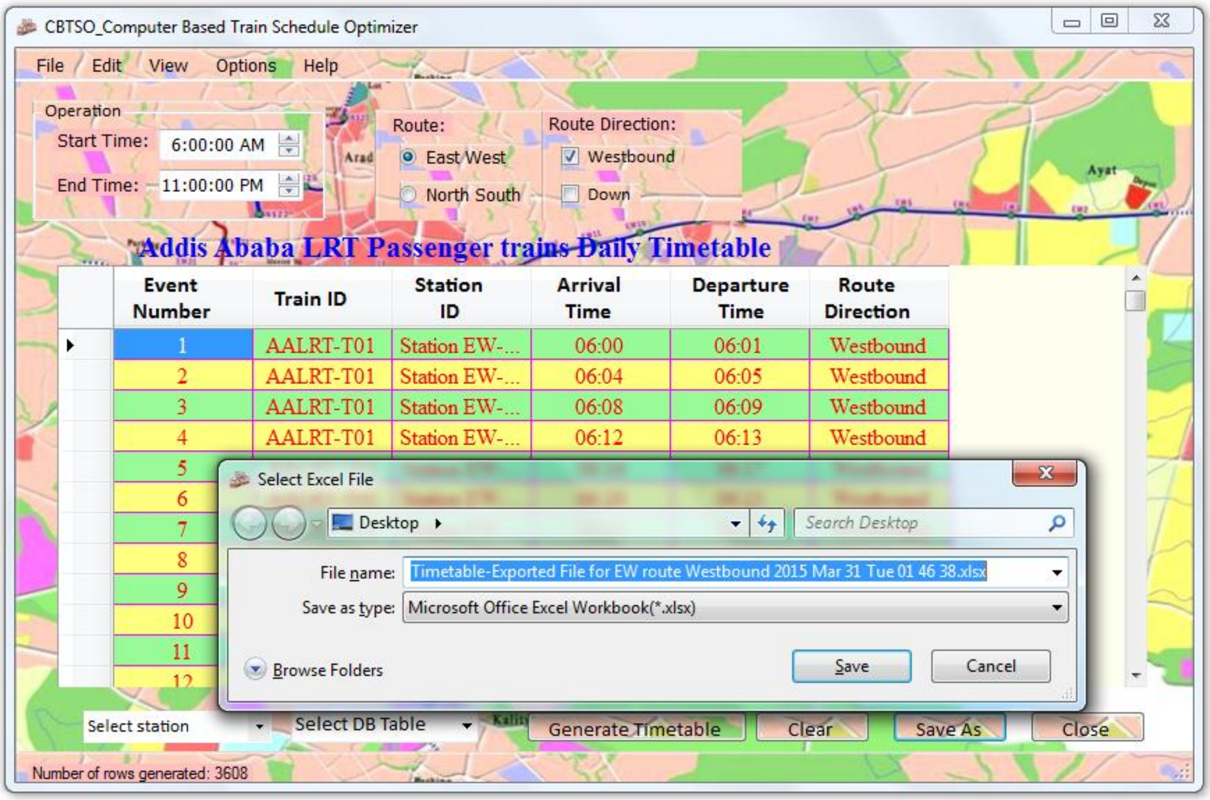


Figure 6.16: LRT train details



TrackID	RouteID	FromStation	ToStation	TrackLength	Remark
12	1	EW11	EW12	740	Other section
13	1	EW12	EW13	920	Other section
14	1	EW13	EW14	725	Other section
15	1	EW14	EW15	565	Other section
16	1	EW15	16	673	Shared line
17	1	16	17	506	Shared line
18	1	17	18	500	Shared line
19	1	18	19	696	Shared line
20	1	19	20	758	Shared line
21	1	20	EW21	696	Other section
22	1	EW21	EW22	1101	Other section
23	1	EW22	Terminal	164	Other section

Figure 6.17: CBTSO track segment details



Event Number	Train ID	Station ID	Arrival Time	Departure Time	Route Direction
1	AALRT-T01	Station EW-...	06:00	06:01	Westbound
2	AALRT-T01	Station EW-...	06:04	06:05	Westbound
3	AALRT-T01	Station EW-...	06:08	06:09	Westbound
4	AALRT-T01	Station EW-...	06:12	06:13	Westbound

Figure 6.18 Saving the generated timetables as excel file by clicking on the save as button

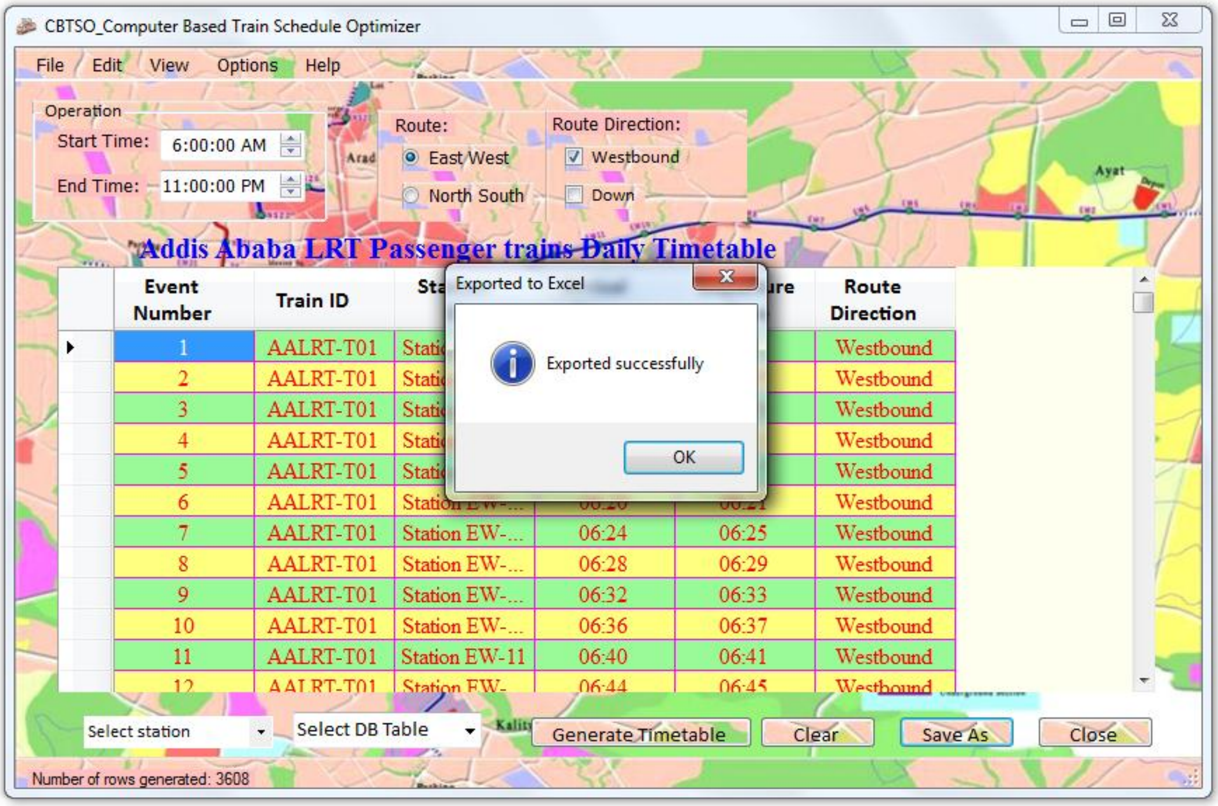
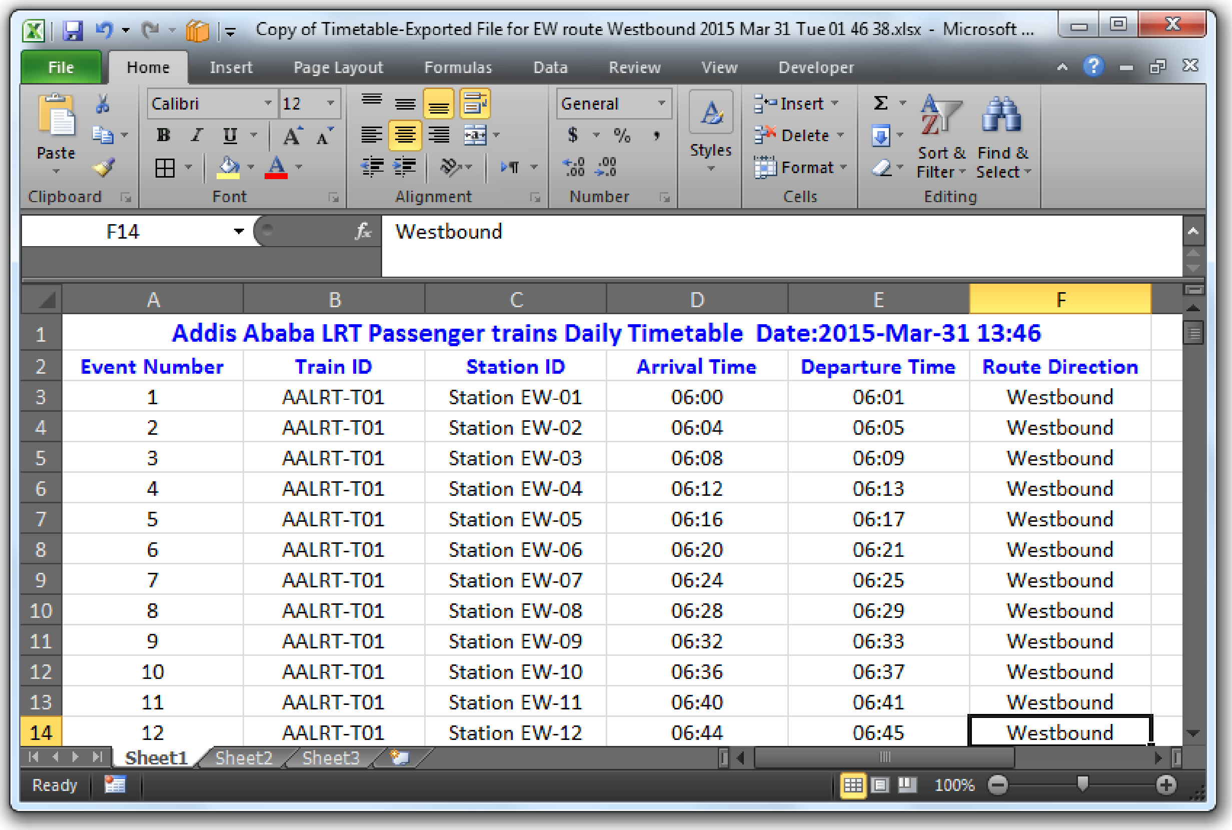
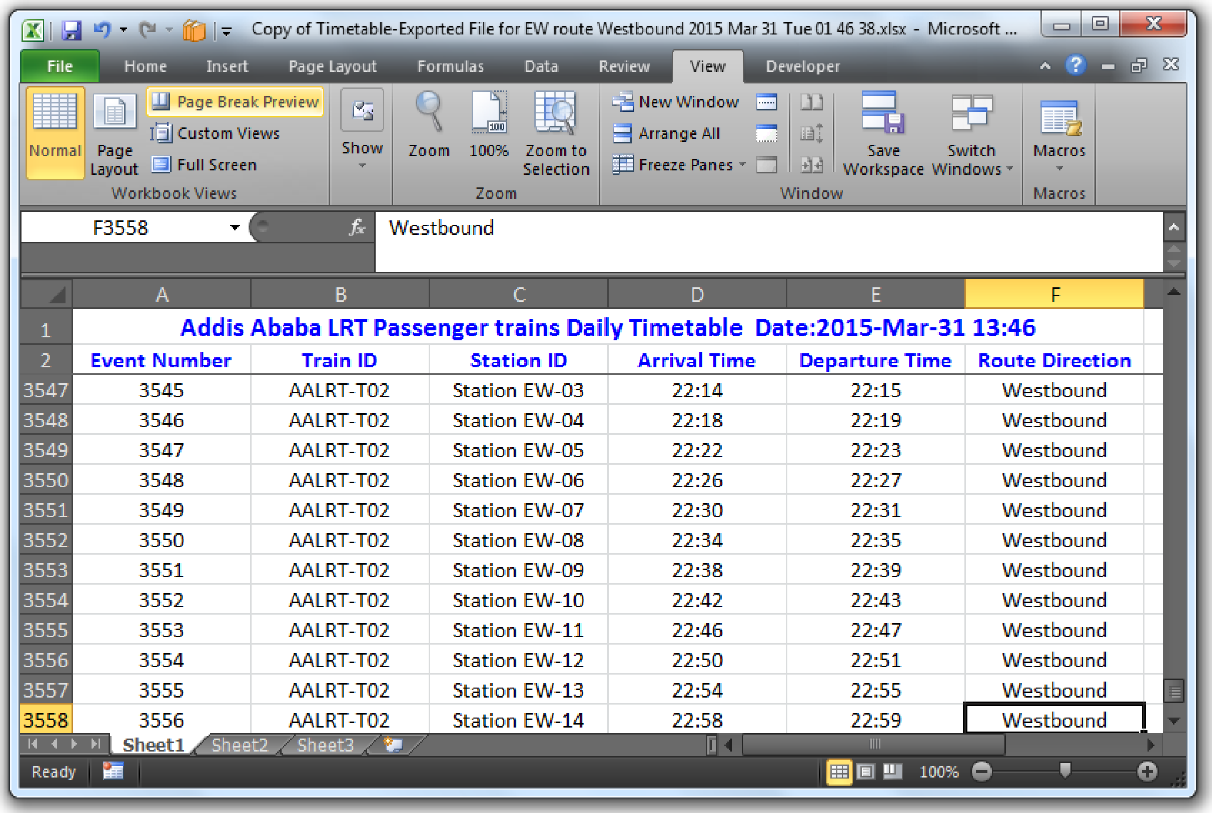


Figure 6.19 Timetable file export successfulness message



a. Initial view the exported timetable



	A	B	C	D	E	F
1	Addis Ababa LRT Passenger trains Daily Timetable Date:2015-Mar-31 13:46					
2	Event Number	Train ID	Station ID	Arrival Time	Departure Time	Route Direction
3547	3545	AALRT-T02	Station EW-03	22:14	22:15	Westbound
3548	3546	AALRT-T02	Station EW-04	22:18	22:19	Westbound
3549	3547	AALRT-T02	Station EW-05	22:22	22:23	Westbound
3550	3548	AALRT-T02	Station EW-06	22:26	22:27	Westbound
3551	3549	AALRT-T02	Station EW-07	22:30	22:31	Westbound
3552	3550	AALRT-T02	Station EW-08	22:34	22:35	Westbound
3553	3551	AALRT-T02	Station EW-09	22:38	22:39	Westbound
3554	3552	AALRT-T02	Station EW-10	22:42	22:43	Westbound
3555	3553	AALRT-T02	Station EW-11	22:46	22:47	Westbound
3556	3554	AALRT-T02	Station EW-12	22:50	22:51	Westbound
3557	3555	AALRT-T02	Station EW-13	22:54	22:55	Westbound
3558	3556	AALRT-T02	Station EW-14	22:58	22:59	Westbound

b. End view of the exported timetable

Figure 0.20 The exported timetable (you can see the partial content of the whole exported time table form the appendix; it's very large collection of records to include the whole content here)

5.3 RESULTS AND INTERPRETATIONS

CBTSO scheduling software can support AA-LRT rail network planners to make daily passenger train schedule with a user friendly GUI interface and it has also a functionality of exporting the timetable table generated by CBTSO as excel file so that the users can utilize all the magical functionalities offered by Microsoft Excel to sort and/or filter by station, route directions, train etc.

5.4 CONCLUSION AND RECOMMENDATIONS

Since the time allowed to complete this research work was shorter than the curriculum allows for research work it is very challenging to deploy a software that can handle all the operational constraints specially the unintended disturbances like delay or emergencies like deadlock where a train can block a track segment due to failure to travel by its planned schedule. Hence, I recommend to all those who are interested to do their research work on this area to focus on how to handle disturbances of train operation. Further I recommend ERC to implement a fully operational web based train scheduling application, using this research work as a prototype that could run on a server and can be accessed from any office of the company.

5.5 REFERENCES

[1]. Arianna Alfieri, Rutger Groot, Leo Kroon, Lex Schrijver (2002). "Efficient Circulation of Railway Rolling Stock". ERIM Report Series Research in Management.

[2]. Banafsheh Khosravi (2013). "Train Scheduling with Application to the UK Rail Network". University of Southampton School of Management. .

[3]. Floudas and P. M. Pardalos (2009). "Encyclopedia of Optimization". Springer Science+BuisinessMedia, LLC.

[4]. C. Barnhart and G. Laporte (2007). "Handbook in Operations Research & Management Science". Elsevier B.V.

[5]. Carey, M. (1998). "Optimizing scheduled times, allowing for behavioral response, Transportation Research". . Part B. Vol.32, No.5, 329-342.

[6]. Chang, Y. H., Yeh, C. H. and Shen, C. C. (2000). "A Multi objective model for passenger train services planning: Application to Taiwan's high-speed rail line, Transportation Research". . Part B, Vol. 34, No. 2, 91-106.

[7]. China Railway Group Limited (CREC) (2009). "Addis Ababa LRT Project, EW & NS Line, Project Study Report, and Addis Ababa E-W & N-S Phase I Light Rail Transit Project EPC Schematic Design".

[8]. Der-San Chen, Robert G. Batson, and Yu Dang (2010). "Applied Integer Programming: Modeling and Solution". John Wiley & Sons, Inc.

[9]. E.M. van Duinkerken (2005). "Modelling Railway Infrastructure Constraints in Max-Plus Algebra". Master Thesis, Delft University of Technology Faculty of Electrical Engineering, Mathematics and Computer Science. .

[10]. European Scene: Support of the Research Process (2007). "Special: Traffic Planning and Logistics". European Research Consortium for Informatics and Mathematics.

[11]. Gabrio Curzio Caimi Dipl. Math. Eth (2009). "Algorithmic Decision Support for Train Scheduling in a Large and Highly Utilised Railway Network". ETH Zurich.

[12]. ***Gebreaftse Gebremichael*** (2014). "Train Scheduling for Addis Aaba Light Rail Transit System". Addis Ababa University Institute of Technology, M.Sc. Thesis.

[13]. Ghoseiri, K., Szidarovszky, F. and Asaharpour M. J. (2004). " A multi-objective train scheduling model and solution, Transportation Research ". Part B, Vol. 38, No. 10, 927-952.

[14]. Gilbert Laporte HEC Montreal (2006). "Handbooks in Operations Research and Management Science, Vol. 14". Cynthia Barnhart Massachusetts Institute of Technology, Elsevier. pp. 69-188.

[15]. Guven Sahin (2006). "New Combinatorial Approaches for Solving Railroad Planning and Scheduling Problems". Phd Dissertation, University of Florida.

[16]. Higgins, A., Kozan, E. and Ferreira, L. (1994b) " Development of a Model to Optimise the Position of Crossing and Overtaking Loops on a Single Line Railway", Physical Infrastructure Research Report, 16-94, Queensland University of Technology, Brisbane

[17]. IBM Corp. 1987, 2011 (2011). "IBM ILOG CPLEX Optimization Studio Getting Started with CPLEX". IBM Corporation.

[18]. IBM ILOG OPL (2009). "Optimization modeling with IBM ILOG OPL: Instructor Workbook". IBM Corporation.

[19]. **Ingo Arne Hansen. Jorn Pachl** (2006). "RAILWAY TIMETABLE & TRAFFIC Analysis · Modelling · Simulation". Eurail Press; Technische Universität Braunschweig & University of Technology Delft.

[20]. Istvan Novak.et. al. (2010). VISUAL STUDIO® 2010 AND .NET 4 SIX-IN-ONE. Wiley Publishing, Inc., Indianapolis, Indiana

[21]. Jason Price, Mike Gunderloy (2002). Mastering Visual C#.NET. San Francisco London. SYBEX Inc., 1151 Marina Village Parkway, Almeda, CA 94501. www.Sybex.com

[22]. Joseph Y-T. Leung (2004). "Handbook of Scheduling: Algorithms, Models, and Performance Analysis". CRC Press LLC.

[23]. Jovanovic, D., and Harker, P.T. (1991). "Tactical scheduling of rail operations: the SCAN-I system, Transportation Science". Vol.25, 46-64.

[24]. Julie Jespersen Groth (2008). PhD Thesis. "Decision Support for the Rolling Stock Dispatcher". Technical University of Denmark Informatics and Mathematical Modelling. .

[25]. Kraay, D. and Harker, P.T. (1995). " Real time scheduling of freight railroads, Transportation Research". . Part B, Vol. 29B, No. 3, 91-107.

[26]. Kraay, D., Harker, P. and Chen, B. (1991) "Optimal Pacing of Trains in Freight Railroads" Operations Research, Vol 39, 82 – 99.

[27]. Kraft, E., R. (1987) "A Branch and Bound Procedure for Optimal Train Dispatching" Journal of the Transportation Research Forum, Vol 28, 263 – 276.

[28]. Leon Willem Petrus Peeters (2003). "Cyclic Railway Timetable Optimization". TRAIL Thesis series T3003/5, TRAIL Research School, Netherlands.

[29]. Liu, Shi Qiang & Kozan, Erhan (2009). "Scheduling trains as a blocking parallel-machine job shop scheduling problem". Queenslan University of Technology Brisbane Australia, Elsevier. http://eprints.qut.edu.au/29818/.

[30]. Mills, R., G., Perkins, S., E. and Pudney, P., J. (1991) "Dynamic Rescheduling of Long Haul Trains for Improved Timekeeping and Energy", Asia-Pacific Journal Operational Research, Vol 8, 146 - 165

[31]. M.R. Bussieck, T.Winter, U.T.Zimmermann (1997). "Discrete Optimization in Public Rail Transport ". http://www.math.tu-bs.de/mo/ismp.html. Journal of the Eastern Asia Society for Transportation Studies, Vol. 6, pp. 372 - 384.

[32]. Marin, A. and Salmerón, J. (1996). "Tactical design of rail freight networks – Part I: Exact and heuristic methods, European Journal of Operation Research". . Vol.90, 26-44.

[33]. Mees, A., I. (1991) "Railway Scheduling by Network Optimisation" Mathematical Computer Modelling, Vol 15, 33 – 41.

[34]. Morlok, E. K. and Peterson, R. B. (1970). "Final report on a development of a geographic transportation network generation and evaluation model". . J. Transportation Research Forum 11, 71-105.

[35]. Newman, A. and Yano, C. A. (2000). "Centralized and decentralized scheduling for intermodal operations". IIE Transactions 32, 743-754.

[36]. Nozick, L. K. and Morlok, E. K. (1997). "A model for medium-term operations planning in an intermodal rail-truck service". . Transportation Research Part A, 91-107.

[37]. Paolo Toth & Daniele Vigo (2002). "The Vehicle Routing Problem". Society for Industrial and Applied Mathematics. Bologna University, Italy.

[38]. Paul Weinberg James Groff Andrew Oppel (2010). SQL the Complete Reference, Third Edition. The McGraw-Hill Companies.

[39]. Petersen, E., R., Taylor, A., J. and Martland, C., D. (1986) "An Introduction to Computer Assisted Train Dispatch", Journal of Advanced Transportation, Vol 20, 63 – 72.

[40]. Prof. Dr.h.c. mult. Martin Grötschel (1905). "Models For The Train Timetabling Problem". vorgelegt von Berkan Erol am Institut für Mathematik der Technischen Universität Berlin. .

[41]. Robert Geisberger (2011). " Advanced Route Planning in Transportation Networks". Von der Fakultät für Informatik des Karlsruher Instituts für Technologie.

[42]. Springer (2004). Science + Business Media, Inc. (2004). "Evolutionary Computation in Combinatorial Optimization: 4th European Conference, EvoCOP 2004 Coimbra, Portugal, April 2004". Springer.

[43]. Szpigel, B. (1973) "Optimal Train Scheduling on a Single Line Railway", Operations Research 72, 344 – 351.

[44]. Thomas Lindner (2000). " Train Schedule Optimization in Public Rail Transport; Thesis". Vom Fachbereich f¨ur Mathematik und Informatik. .

[45]. Tomas Robenek,Jianghang Chen,Michel Bierlaire (2013). "Exploratory Analysis of Demand Interaction in the Planning of Passenger Railway Service". Transp - OR, EPFL, 13th Swiss Transport Research Conference.

[46]. Vincent T'kindt Jean-Charles Billaut (2006). " Mult criteria Scheduling Theory, Models and Algorithms". Springer-Verlag Berlin Heidelberg Printed in Germany.

[47]. Wilson, Ihrig & Associates, Inc. Emeryville, CA (2012). "Track Design Handbook for Light Rail Transit Second Edition". Parsons Brinckerhoff, Inc. WASHINGTON, D.C.

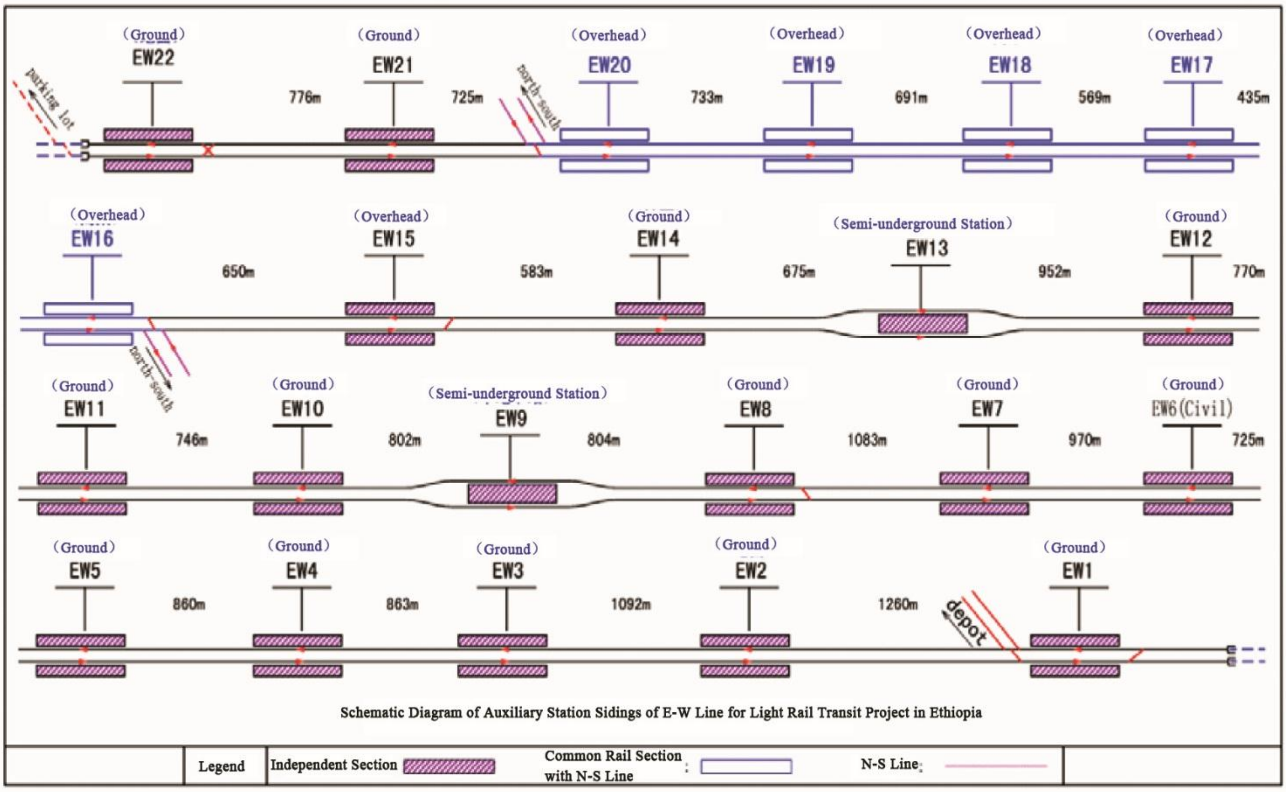
[48]. Elias Silva de Oliveira (2001) « "Solving Single Track Railway Scheduling Problem Using Constraint Programming", Phd Thesis, The University of Leeds school of Computing

[49]. Van Hentenryck, P. and Saraswat, V. (1997) "Constraint Programming: Strategic Direction. Constraints: An International Journal" 2(1):7-33.

[50]. Brailsford, Sally C., Potts, C.N., and Smith, Barbara M. (1999)"Constraint Satisfaction Problems: Algorithms. European Journal of Operational Research", (119): 557-581.

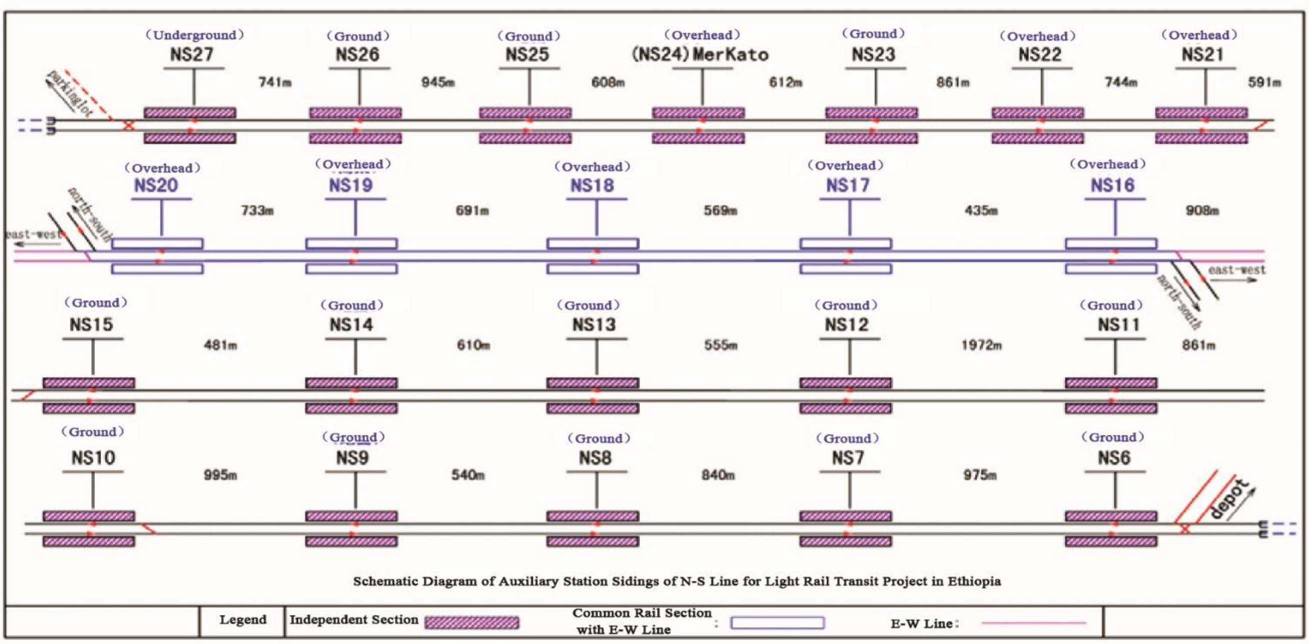
1 ,APPENDIX

APPENDIX A



Schematic Diagram of Auxiliary Station Sidings of E-W Line for Light Rail Transit Project in Ethiopia

Figure 6.21: Schematic diagram of station sidings of east-west (E-W) line



Schematic Diagram of Auxiliary Station Sidings of N-S Line for Light Rail Transit Project in Ethiopia

Figure 6.22: Schematic diagram of station sidings of north-south (N-S) line

The table show in the following page contains the distance between each consecutive passenger stops.

from-to	distance(m)	from-to	distance(m)	from-to	distance(m)	from-to	distance(m)
						EW16-EW17	435
EW1-EW2	1260	EW6-EW7	970	EW11-EW12	770	EW17-EW18	569
EW2-EW3	1092	EW7-EW8	1083	EW12-EW13	952	EW18-EW19	691
EW3-EW4	863	EW8-EW9	804	EW13-EW14	675	EW19-EW20	733
EW4-EW5	860	EW9-EW10	802	EW14-EW15	583	EW20-EW21	725
EW5-EW6	725	EW10-EW11	746	EW15-EW16	650	EW21-EW22	776
Sub Total_1	4800	Sub Total_2	4405	Sub Total_3	3630	Sub Total_4	3929
Total Distance in the East-West Corridor= $\sum_{i=1}^{4} Sub\ Total\_i =$ 16764 meteres							
from-to	distance(m)	from-to	distance(m)	from-to	distance(m)	from-to	distance(m)
						NS21-NS22	744
NS6-NS7	975	NS11-NS12	1972	NS16-NS17	435	NS22-NS23	861
NS7-NS8	840	NS12-NS13	555	NS17-NS18	569	NS23-NS24	612
NS8-NS9	540	NS13-NS14	610	NS18-NS19	691	NS24-NS25	608
NS9-NS10	995	NS14-NS15	481	NS19-NS20	733	NS25-NS26	945
NS10-NS11	861	NS15-NS16	908	NS20-NS21	591	NS26-NS27	741
Sub Total_1	4211	Sub Total_2	4526	Sub Total_3	3019	Sub Total_4	4511
Total Distance in the East-West Corridor = $\sum_{i=1}^{4} Sub\ Total\_i =$ 16267 meteres							
Shared Rail = EW16 to EW20 = NS16 to NS20 = 2428 meters							

Table 7.1 Tabular form of station to station distances

CHAPTER 7

B.1 SOURCE CODE FOR CBTSO TRAIN TIMETABLE TOOL

```
/*-------------------------------------------------------------------------------
<Train timetable optimization source code for Addis Ababa LRT System>
Author: Tesfaye Belachew
Email: tesfanurr@gmail.com
Railway Electrical Engineering masters 2nd batch student
Institute: Addis Ababa Institute of Technology/Railway Institute
Year: 2014/2015
</Train timetable optimization source for Addis Ababa LRT System>
---------------------------------------------------------------------------
*/

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;//added to handle the case of linked list collection
using System.Text;
using System.Windows.Forms;
using System.Data.Sql;
using System.Data.SqlClient;//Sql server data provider
using System.Reflection;
using Excel = Microsoft.Office.Interop.Excel;//added to handle excel file export functionality


namespace CBTSO
{
public partial class formCBTSO : Form
{
public formCBTSO()
{
InitializeComponent();
}


private void exitToolStripMenuItem_Click(object sender, EventArgs e)
{
Close();
}


private void radiobtn_EastWest_CheckedChanged(object sender, EventArgs e)
{

}

private void btnClose_Click(object sender, EventArgs e)
{
this.Close();
}


private void fillByToolStripButton_Click(object sender, EventArgs e)
{
try
{
this.trackSegmentTableAdapter.FillBy(this.aA_LRTDataSet1.TrackSegment);
}
catch (System.Exception ex)
```

```
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}

}

private void fillToolStripButton_Click(object sender, EventArgs e)
{
try
{
this.stationTableAdapter.Fill(this.aA_LRTDataSet1.Station);
}
catch (System.Exception ex)
{
System.Windows.Forms.MessageBox.Show(ex.Message);
}
}
private void button1_Click(object sender, EventArgs e)
{
    try
    {
    this.stationTableAdapter.Fill(this.aA_LRTDataSet1.Station);
    }
    catch (System.Exception ex)
    {
    System.Windows.Forms.MessageBox.Show(ex.Message);
    }
}

private void buttonFillData_Click(object sender, EventArgs e)
{
string connectionString = @"Data Source=.\SQLEXPRESS;Initial Catalog = AA_LRT;Integrated Security=True";
string sql = "SELECT * FROM Timetable";
SqlConnection connection = new SqlConnection(connectionString);
SqlDataAdapter dataadapter = new SqlDataAdapter(sql, connection);
DataSet ds = new DataSet();
connection.Open();
dataadapter.Fill(ds, "Timetable");
connection.Close();
if (this.dataGridView1.DataSource != null)
{
this.dataGridView1.DataSource = null;
}
else
{
this.dataGridView1.Rows.Clear();
}
dataGridView1.DataSource = ds;
dataGridView1.DataMember = "Timetable";

}

private void comboBoxDbTable_SelectedValueChanged(object sender, EventArgs e)
{

string connectionString = @"Data Source=.\SQLEXPRESS;Initial Catalog=AA_LRT;Integrated Security=True";

string Table = comboBox1.Text.ToString();


if (Table == "OperationDay")
{

string sql = "SELECT * FROM OperationDay";
SqlConnection connection = new SqlConnection(connectionString);
```

```
SqlDataAdapter dataadapter = new SqlDataAdapter(sql, connection);
DataSet ds = new DataSet();
connection.Open();
dataadapter.Fill(ds, "OperationDay");
connection.Close();
if (this.dataGridView1.DataSource != null)
{
this.dataGridView1.DataSource = null;
}
else
{
this.dataGridView1.Rows.Clear();
}
dataGridView1.DataSource = ds;
dataGridView1.DataMember = "OperationDay";
dataGridView1.DefaultCellStyle.BackColor = Color.PaleGreen;
}
else if (Table == "Route")
{
string sql = "SELECT * FROM Route";
SqlConnection connection = new SqlConnection(connectionString);
SqlDataAdapter dataadapter = new SqlDataAdapter(sql, connection);
DataSet ds = new DataSet();
connection.Open();
dataadapter.Fill(ds, "Route");
connection.Close();
if (this.dataGridView1.DataSource != null)
{
this.dataGridView1.DataSource = null;
}
else
{
this.dataGridView1.Rows.Clear();
}
dataGridView1.DataSource = ds;
dataGridView1.DataMember = "Route";

}

else if (Table == "Station")
{
string sql = "SELECT * FROM Station";
SqlConnection connection = new SqlConnection(connectionString);
SqlDataAdapter dataadapter = new SqlDataAdapter(sql, connection);
DataSet ds = new DataSet();
connection.Open();
dataadapter.Fill(ds, "Station");
connection.Close();
if (this.dataGridView1.DataSource != null)
{
this.dataGridView1.DataSource = null;
}
else
{
this.dataGridView1.Rows.Clear();
}
dataGridView1.DataSource = ds;
dataGridView1.DataMember = "Station";
}
else if (Table == "Timetable")
{
string sql = "SELECT * FROM Timetable";
SqlConnection connection = new SqlConnection(connectionString);
SqlDataAdapter dataadapter = new SqlDataAdapter(sql, connection);
DataSet ds = new DataSet();
```

```
connection.Open();
dataadapter.Fill(ds, "Timetable");
connection.Close();
if (this.dataGridView1.DataSource != null)
{
this.dataGridView1.DataSource = null;
}
else
{
this.dataGridView1.Rows.Clear();
}
dataGridView1.DataSource = ds;
dataGridView1.DataMember = "Timetable";
dataGridView1.DefaultCellStyle.BackColor = Color.PaleGreen;
}
else if (Table == "TrackSegment")
{
string sql = "SELECT * FROM TrackSegment";
SqlConnection connection = new SqlConnection(connectionString);
SqlDataAdapter dataadapter = new SqlDataAdapter(sql, connection);
DataSet ds = new DataSet();
connection.Open();
dataadapter.Fill(ds, "TrackSegment");
connection.Close();
if (this.dataGridView1.DataSource != null)
{
this.dataGridView1.DataSource = null;
}
else
{
this.dataGridView1.Rows.Clear();
}
dataGridView1.DataSource = ds;
dataGridView1.DataMember = "TrackSegment";
dataGridView1.DefaultCellStyle.BackColor = Color.PaleGreen;
}
else if (Table == "Train")
{
string sql = "SELECT * FROM Train";
SqlConnection connection = new SqlConnection(connectionString);
SqlDataAdapter dataadapter = new SqlDataAdapter(sql, connection);
DataSet ds = new DataSet();
connection.Open();
dataadapter.Fill(ds, "Train");
connection.Close();
if (this.dataGridView1.DataSource != null)
{
this.dataGridView1.DataSource = null;
}
else
{
this.dataGridView1.Rows.Clear();
}
dataGridView1.DataSource = ds;
dataGridView1.DataMember = "Train";
dataGridView1.DefaultCellStyle.BackColor = Color.PaleGreen;
}
}
private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
}
private void comBox_SelectStation_SelectedIndexChanged(object sender, EventArgs e)
{
MessageBox.Show(comBox_SelectStation.Text);
}
```

```
// Export results into excel file
private void ExportAsExcelFile()
{
Microsoft.Office.Interop.Excel.Application excelApp = null;
try
{
// instantiating the excel application class
object misValue = System.Reflection.Missing.Value;
excelApp = new Microsoft.Office.Interop.Excel.Application();
Microsoft.Office.Interop.Excel.Workbook currentWorkbook = excelApp.Workbooks.Add(Type.Missing);
Microsoft.Office.Interop.Excel.Worksheet currentWorksheet = (Microsoft.Office.Interop.Excel.Worksheet)currentWorkbook.ActiveSheet;
currentWorksheet.Columns.ColumnWidth = 15;
if (dataGridView1.Rows.Count > 0)
{
currentWorksheet.Range[currentWorksheet.Cells[1, 1], currentWorksheet.Cells[1, 6]].Merge();
currentWorksheet.Cells[1,1] = labelMessage.Text + "  "+"Date:"+""+ DateTime.Now.ToString("yyy-MMM-dd HH:mm");
currentWorksheet.Range["A1:G1"].Font.Bold = true;
currentWorksheet.Range["A1:G1"].Font.Size = 14;
currentWorksheet.Range["A1:G1"].Font.Color = 0xFF0000;

int i = 1;
foreach (DataGridViewColumn dgviewColumn in dataGridView1.Columns)
{
// Excel work sheet indexing starts with 1
currentWorksheet.Cells[2, i] = dgviewColumn.Name;
++i;
}
Microsoft.Office.Interop.Excel.Range headerColumnRange = currentWorksheet.get_Range("A2", "I2");
headerColumnRange.Font.Bold = true;
headerColumnRange.Font.Color = 0xFF0000;
//headerColumnRange.EntireColumn.AutoFit();
int rowIndex = 0;
for (rowIndex = 0; rowIndex < dataGridView1.Rows.Count; rowIndex++)
{
DataGridViewRow dgRow = dataGridView1.Rows[rowIndex];
for (int cellIndex = 0; cellIndex < dgRow.Cells.Count; cellIndex++)
{
currentWorksheet.Cells[rowIndex + 3, cellIndex + 1] = dgRow.Cells[cellIndex].Value;
}
}
Microsoft.Office.Interop.Excel.Range fullTextRange = currentWorksheet.get_Range("A1", "G" + (rowIndex + 1).ToString());
fullTextRange.WrapText = true;
fullTextRange.HorizontalAlignment = Microsoft.Office.Interop.Excel.XlHAlign.xlHAlignLeft;
}
else
{// try find the header for the conetent of the exported excel in the first row.
string timeStamp = DateTime.Now.ToString("s");
timeStamp = timeStamp.Replace(':', '-');
timeStamp = timeStamp.Replace("T", "__");
currentWorksheet.Cells[1, 1] = timeStamp;
currentWorksheet.Cells[1, 2] = "No error occured";
}
using (SaveFileDialog exportSaveFileDialog = new SaveFileDialog())
{
string format3 = "yyyy MMM dd ddd hh mm ss";

exportSaveFileDialog.Title = "Select Excel File";
exportSaveFileDialog.FileName = "Exported File" + " " + DateTime.Now.ToString(format3);//.Replace('/', '-').Replace(':', ' ');

exportSaveFileDialog.Filter = "Microsoft Office Excel Workbook(*.xlsx)|*.xlsx";
```

```
if (DialogResult.OK == exportSaveFileDialog.ShowDialog())
{
string fullFileName = exportSaveFileDialog.FileName;
//currentWorkbook.SaveCopyAs("fullFileName");
// indicating that we already saved the workbook, otherwise call to Quit() will pop up
// the save file dialogue box
urrentWorkbook.SaveAs(fullFileName, Microsoft.Office.Interop.Excel.XlFileFormat.xlOpenXMLWorkbook,
System.Reflection.Missing.Value, misValue, false, false,
Microsoft.Office.Interop.Excel.XlSaveAsAccessMode.xlNoChange,
Microsoft.Office.Interop.Excel.XlSaveConflictResolution.xlUserResolution, true, misValue, misValue, misValue);
currentWorkbook.Saved = true;
MessageBox.Show("Exported successfully", "Exported to Excel", MessageBoxButtons.OK,
MessageBoxIcon.Information);
}
}
}
catch (Exception ex)
{
MessageBox.Show(ex.Message, "Exception", MessageBoxButtons.OK, MessageBoxIcon.Error);
}
finally
{
if (excelApp != null)
{
excelApp.Quit();
}
}
}
private void buttonExportToExcelFile_Click(object sender, EventArgs e)
{
ExportAsExcelFile();
}

private void label2_Click(object sender, EventArgs e)
{
}
private void buttonClear_Click(object sender, EventArgs e)
{
if (this.dataGridView1.DataSource != null)
{
this.dataGridView1.DataSource = null;
}
else
{
this.dataGridView1.Rows.Clear();
}
}
private void buttonGenerateTimtable_Click(object sender, EventArgs e)
{
MessageBox.Show("Please choose a specific route, route direction and the station first!\n Thank you for using
CBTSO!!");
//After the above message has been diplayed for the end user, they should click the ok button to return back to
//selection and you code should perform that operation fo the end users.
string connectionString;
connectionString = @"Data Source= .\SQLEXPRESS;Initial Catalog=AA_LRT;Integrated Security=True";
// string sql = "SELECT * FROM Timetable";
string sql = "SELECT TrainID,StationID,ArrivalTime,DepartureTime FROM Timetable";
SqlConnection connection = new SqlConnection(connectionString);
SqlDataAdapter dataadapter = new SqlDataAdapter(sql, connection);
DataSet ds = new DataSet();
connection.Open();
dataadapter.Fill(ds, "Timetable");
connection.Close();
if (this.dataGridView1.DataSource != null)
{
```

```
this.dataGridView1.DataSource = null;
}
else
{
this.dataGridView1.Rows.Clear();
}
dataGridView1.DataSource = ds;
dataGridView1.DataMember = "Timetable";
}
private void enterScheduleParametersToolStripMenuItem_Click(object sender, EventArgs e)
{
FormInputParameters InputParameters = new FormInputParameters();
InputParameters.Show();
}
private void btnBuildSchedule(object sender, EventArgs e)
{
labelMessage.Text = "Addis Ababa LRT Passenger trains Daily Timetable";
//..................................................................
//start Schedule for EW route
//..................................................................
if ((this.rdBtnEastWest.Checked && checkBoxUp.Checked) || (this.rdBtnEastWest.Checked &&
checkBoxDown.Checked))
{
//Clear dataGridView before excuting the schedule
if (this.dataGridView1.DataSource != null)
{
this.dataGridView1.DataSource = null;
}
else
{
this.dataGridView1.Rows.Clear();
}
checkBoxUp.Text = "Westbound";
//-----------------------------
//string format1 = "MMM ddd d HH:mm yyyy";
//string format = " ddd-dd-MMM-yyyy HH:mm ";
//string format = "yyyy HH:mm"; //24 hour format
string twentyfourHourFormat = "HH:mm"; //24 hour format
string twoDigitFormat = "D2";
dataGridView1.ColumnCount = 6;
dataGridView1.Columns[0].Name = "Event Number";
dataGridView1.Columns[1].Name = "Train ID";
dataGridView1.Columns[2].Name = "Station ID";
dataGridView1.Columns[3].Name = "Arrival Time";
dataGridView1.Columns[4].Name = "Departure Time";
dataGridView1.Columns[5].Name = "Route Direction";
dataGridView1.DefaultCellStyle.BackColor = Color.PaleGreen;

DateTime[,] scheduleArrivTime = new DateTime[50, 50];//[trainIndex, stationIndex]
DateTime[,] scheduleDepTime = new DateTime[50, 50];
//3 dimentional array length
//scheduleDepTime.GetLength(0);//43
//scheduleDepTime.GetLength(1);//44
//scheduleArrivTime.GetLength(0);//45
//scheduleArrivTime.GetLength(1);//46

DateTime startTime = dTPStartTime.Value;//starting time of daily trains operation
DateTime EndTime = dTPEndTime.Value;
int eventNumber = 1;
for (int trainIndex = 1; trainIndex <= 41; trainIndex++)
{
for (int stationIndex = 1; stationIndex <= 44; stationIndex++)
{

string[] row = new string[] { trainIndex.ToString(), "Arrival Time", "Departure Time" };
```

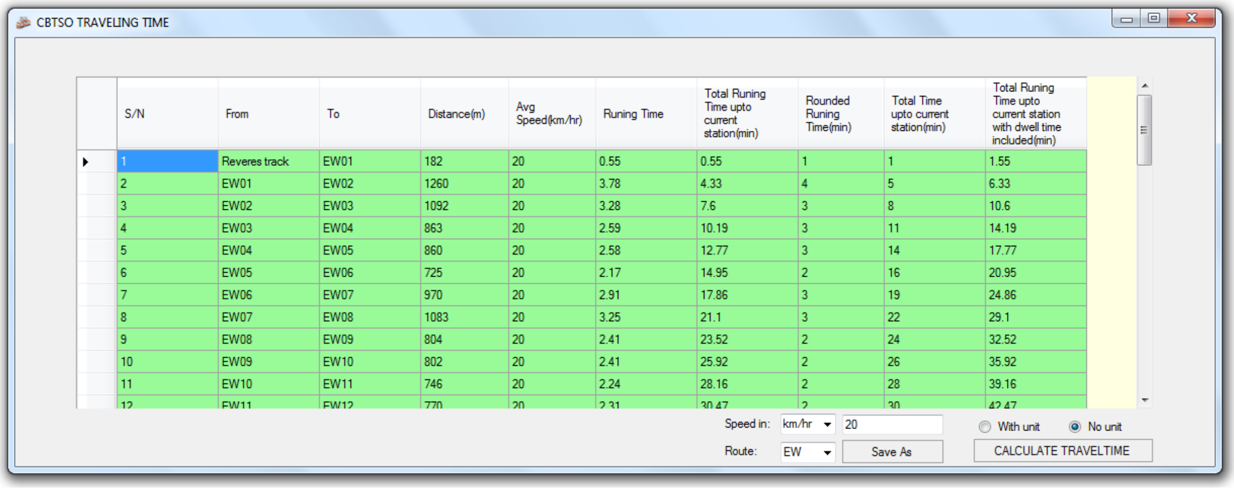
```
//you can make the following variables to set by the user themselves from  the GUI windows from controls like the textBoxes
int dwellTime = 1;//2min dwell time at each station
int runningTime = 3;//3min running time between station
int headwayTimeForOtherTrackSection = 6;// time interval between two trains
//// int headwayTimeForSharedTrack = 3;
//set arrival and departure time for the first train
if (stationIndex == 1 && trainIndex == 1)
{
//scheduleArrivTime[1, 1] = startTime or arrival time for the first train at the first station
scheduleArrivTime[stationIndex, trainIndex] = startTime;
scheduleDepTime[stationIndex, trainIndex] = scheduleArrivTime[stationIndex, trainIndex].AddMinutes(dwellTime);
row = new string[] {eventNumber.ToString(), "AALRT-T"+ trainIndex.ToString(twoDigitFormat), "Station EW-"+ (stationIndex).ToString(twoDigitFormat),
scheduleArrivTime[stationIndex,trainIndex].ToString("HH:mm"), scheduleDepTime[stationIndex, trainIndex].ToString(twentyfourHourFormat),"Westbound" };

if (this.rdBtnEastWest.Checked && checkBoxUp.Checked)
{
dataGridView1.Rows.Add(row);
labelNumberofRowsGenerated.Text = "Number of rows generated: " + eventNumber.ToString();
}
eventNumber++;
}

//verify safety constraints
else if (trainIndex > 1 && stationIndex == 1)
{
scheduleArrivTime[stationIndex, trainIndex] = startTime.AddMinutes((trainIndex)* headwayTimeForOtherTrackSection);//arrival time for the rest of the train at the first station

//scheduleArrivTime[stationIndex, myTrainID + 1] = scheduleArrivTime[stationIndex, myTrainID].AddMinutes(headwayTime);//arrival headway time
scheduleDepTime[stationIndex, trainIndex] = scheduleArrivTime[stationIndex, trainIndex].AddMinutes((trainIndex) * headwayTimeForOtherTrackSection);//departure headway time
//scheduleArrivTime[stationIndex, myTrainID + 1] = scheduleArrivTime[stationIndex, myTrainID].AddMinutes(headwayTime);
row = new string[] {eventNumber.ToString(), "AALRT-T"+ trainIndex.ToString(twoDigitFormat), "Station EW-" + (stationIndex).ToString(twoDigitFormat), scheduleArrivTime[stationIndex,
trainIndex].ToString(twentyfourHourFormat), scheduleDepTime[stationIndex, trainIndex].ToString(twentyfourHourFormat),"Westbound" };

if (this.rdBtnEastWest.Checked && checkBoxUp.Checked)
{
dataGridView1.Rows.Add(row);
labelNumberofRowsGenerated.Text = eventNumber.ToString();
}
eventNumber++;
}
//for stationIndex > 1
else if ((stationIndex > 1 && stationIndex <= 22) && trainIndex >= 1)
{
//train_i_ArrivTimeAtStation_k+1 = train_i_DepTimeFromStation_k + runningTimeFrom_k_to_k+1
scheduleArrivTime[stationIndex, trainIndex] = scheduleDepTime[stationIndex - 1, trainIndex].AddMinutes(runningTime);
//scheduleArrivTime[stationIndex, myTrainID] = startTime + runningTime;
scheduleDepTime[stationIndex, trainIndex] = scheduleArrivTime[stationIndex, trainIndex].AddMinutes(dwellTime);
//train_i_DepTimeFromStation_k = train_i_DepTimeFromStation_k + dwellTimeAtStation_k
//scheduleArrivTime[stationIndex + 1, myTrainID] = scheduleDepTime[stationIndex, myTrainID].AddMinutes(1);
row = new string[] {eventNumber.ToString(),"AALRT-T"+  trainIndex.ToString(twoDigitFormat), "Station EW-"+ (stationIndex).ToString(twoDigitFormat), scheduleArrivTime[stationIndex,
trainIndex].ToString(twentyfourHourFormat), scheduleDepTime[stationIndex, trainIndex].ToString(twentyfourHourFormat),"Westbound" };

if (this.rdBtnEastWest.Checked && checkBoxUp.Checked)
```

```
{
dataGridView1.Rows.Add(row);
labelNumberofRowsGenerated.Text = "Number of rows generated: " + eventNumber.ToString();
}
eventNumber++;
}
else if (stationIndex > 22 && trainIndex >= 1 && checkBoxDown.Checked)
{
checkBoxDown.Text = "Eastbound";
//train_i_ArrivTimeAtStation_k+1 = train_i_DepTimeFromStation_k + runningTimeFrom_k_to_k+1
scheduleArrivTime[stationIndex, trainIndex] = scheduleDepTime[stationIndex - 1,
trainIndex].AddMinutes(runningTime);
//scheduleArrivTime[stationIndex, myTrainID] = startTime + runningTime;
scheduleDepTime[stationIndex, trainIndex] = scheduleArrivTime[stationIndex, trainIndex].AddMinutes(dwellTime);
//train_i_DepTimeFromStation_k = train_i_DepTimeFromStation_k + dwellTimeAtStation_k
//scheduleArrivTime[stationIndex + 1, myTrainID] = scheduleDepTime[stationIndex, myTrainID].AddMinutes(1);
row = new string[] {eventNumber.ToString(),"AALRT-T"+  trainIndex.ToString(twoDigitFormat), "Station EW-"+ (44-
stationIndex).ToString(twoDigitFormat), scheduleArrivTime[stationIndex,
trainIndex].ToString(twentyfourHourFormat), scheduleDepTime[stationIndex,
trainIndex].ToString(twentyfourHourFormat),"Eastbound" };

if (this.rdBtnEastWest.Checked && checkBoxDown.Checked)
{
dataGridView1.Rows.Add(row);
labelNumberofRowsGenerated.Text = "Number of rows generated: " + eventNumber.ToString();
}
eventNumber++;
}
}//end of station loop
}//end of train loop
}//end of EW Schedule
//................................................................
//Start Schedule for NS route
//................................................................

else if ((this.rdBtnNorthSouth.Checked && checkBoxUp.Checked) || (this.rdBtnNorthSouth.Checked &&
checkBoxDown.Checked))
{
//Clear dataGridView before excuting the schedule
if (this.dataGridView1.DataSource != null)
{
this.dataGridView1.DataSource = null;
}
else
{
this.dataGridView1.Rows.Clear();
}
//----------------------------
checkBoxUp.Text = "Northbound";
//string format1 = "MMM ddd d HH:mm yyyy";
//string format = " ddd-dd-MMM-yyyy HH:mm ";
//string format = "yyyy HH:mm"; //24 hour format
string twentyFourHourFormat = "HH:mm"; //24 hour format
string twoDigitFormat = "D2";
dataGridView1.ColumnCount = 6;
dataGridView1.Columns[0].Name = "Event Number";
dataGridView1.Columns[1].Name = "Train ID";
dataGridView1.Columns[2].Name = "Station ID";
dataGridView1.Columns[3].Name = "Arrival Time";
dataGridView1.Columns[4].Name = "Departure Time";
dataGridView1.Columns[5].Name = "Route Direction";
dataGridView1.DefaultCellStyle.BackColor = Color.PaleGreen;

DateTime[,] scheduleArrivTime = new DateTime[50, 50];//[trainIndex, stationIndex]
DateTime[,] scheduleDepTime = new DateTime[50, 50];
```

```
//3 dimentional array length

DateTime startTime = dTPStartTime.Value;//starting time of daily trains operation
DateTime EndTime = dTPEndTime.Value;
int eventNumber = 1;
for (int trainIndex = 1; trainIndex <= train.Count; trainIndex++)
{
for (int stationIndex = 1; stationIndex <= station.Count; stationIndex++)
{

string[] row = new string[] { trainIndex.ToString(), "Arrival Time", "Departure Time" };
//you can make the following variables to set by the user themselves from  the GUI windows from controls like the
textBoxes
int dwellTime =  Convert.ToInt16(txtDwellTime.Text);;//2min dwell time at each station
int runningTime = getRunningTime();;//3min running time between station
int headwayTimeForOtherTrackSection = 6;// time interval between two trains
//// int headwayTimeForSharedTrack = 3;
//set arrival and departure time for the first train
if (stationIndex == 1 && trainIndex == 1)
{
//scheduleArrivTime[1, 1] = startTime or arrival time for the first train at the first station
scheduleArrivTime[stationIndex, trainIndex] = startTime;
//stop time constraint
scheduleDepTime[stationIndex, trainIndex] = scheduleArrivTime[stationIndex, trainIndex].AddMinutes(dwellTime);
row = new string[] {eventNumber.ToString(), "AALRT-T"+  trainIndex.ToString(twoDigitFormat), "Station NS-"+" "+
(5 + stationIndex).ToString(twoDigitFormat),
scheduleArrivTime[stationIndex,trainIndex].ToString("HH:mm"), scheduleDepTime[stationIndex,
trainIndex].ToString(twentyFourHourFormat),"Northbound" };

if (this.rdBtnNorthSouth.Checked && checkBoxUp.Checked)
{
dataGridView1.Rows.Add(row);
labelNumberofRowsGenerated.Text = "Total number of rows generated: " + eventNumber.ToString();
}
eventNumber++;
}


else if (trainIndex > 1 && stationIndex == 1)
{
scheduleArrivTime[stationIndex, trainIndex] = startTime.AddMinutes((trainIndex)*
headwayTimeForOtherTrackSection);//arrival time for the rest of the train at the first station

//scheduleArrivTime[stationIndex, myTrainID + 1] = scheduleArrivTime[stationIndex,
myTrainID].AddMinutes(headwayTime);//arrival headway time
//safety constraint
scheduleDepTime[stationIndex, trainIndex] = scheduleArrivTime[stationIndex, trainIndex].AddMinutes((trainIndex) *
headwayTimeForOtherTrackSection);//departure headway time
//scheduleArrivTime[stationIndex, myTrainID + 1] = scheduleArrivTime[stationIndex,
myTrainID].AddMinutes(headwayTime);
row = new string[] {eventNumber.ToString(), "AALRT-T"+ trainIndex.ToString(twoDigitFormat), "Station NS-" + (5 +
stationIndex).ToString(twoDigitFormat), scheduleArrivTime[stationIndex,
trainIndex].ToString(twentyFourHourFormat), scheduleDepTime[stationIndex,
trainIndex].ToString(twentyFourHourFormat),"Northbound" };

if (this.rdBtnNorthSouth.Checked && checkBoxUp.Checked)
{
dataGridView1.Rows.Add(row);
labelNumberofRowsGenerated.Text = "Total number of rows generated: " + eventNumber.ToString();
}
eventNumber++;
}
//for stationIndex > 1
else if ((stationIndex > 1 && stationIndex <= 22) && trainIndex >= 1)
{
```

```
//train_i_ArrivTimeAtStation_k+1 = train_i_DepTimeFromStation_k + runningTimeFrom_k_to_k+1
scheduleArrivTime[stationIndex, trainIndex] = scheduleDepTime[stationIndex - 1,
trainIndex].AddMinutes(runningTime);
//scheduleArrivTime[stationIndex, myTrainID] = startTime + runningTime;
scheduleDepTime[stationIndex, trainIndex] = scheduleArrivTime[stationIndex, trainIndex].AddMinutes(dwellTime);
//train_i_DepTimeFromStation_k = train_i_DepTimeFromStation_k + dwellTimeAtStation_k
//scheduleArrivTime[stationIndex + 1, myTrainID] = scheduleDepTime[stationIndex, myTrainID].AddMinutes(1);
row = new string[] {eventNumber.ToString(),"AALRT-T"+ trainIndex.ToString(twoDigitFormat ), "Station NS-"+ (5 +
stationIndex).ToString(twoDigitFormat ), scheduleArrivTime[stationIndex,
trainIndex].ToString(twentyFourHourFormat), scheduleDepTime[stationIndex,
trainIndex].ToString(twentyFourHourFormat),"Northbound" };

if (this.rdBtnNorthSouth.Checked && checkBoxUp.Checked)
{
dataGridView1.Rows.Add(row);
}
eventNumber++;
}
else if (stationIndex > 22 && trainIndex >= 1 && checkBoxDown.Checked)
{
checkBoxDown.Text = "Southbound";
//train_i_ArrivTimeAtStation_k+1 = train_i_DepTimeFromStation_k + runningTimeFrom_k_to_k+1
//Running time constraint
scheduleArrivTime[stationIndex, trainIndex] = scheduleDepTime[stationIndex - 1,
trainIndex].AddMinutes(runningTime);
//scheduleArrivTime[stationIndex, myTrainID] = startTime + runningTime;
scheduleDepTime[stationIndex, trainIndex] = scheduleArrivTime[stationIndex, trainIndex].AddMinutes(dwellTime);
//train_i_DepTimeFromStation_k = train_i_DepTimeFromStation_k + dwellTimeAtStation_k
//scheduleArrivTime[stationIndex + 1, myTrainID] = scheduleDepTime[stationIndex, myTrainID].AddMinutes(1);
row = new string[] {eventNumber.ToString(),"AALRT-T"+ trainIndex.ToString(twoDigitFormat), "Station NS-"+" "+
(49 - stationIndex).ToString(twoDigitFormat), scheduleArrivTime[stationIndex,
trainIndex].ToString(twentyFourHourFormat), scheduleDepTime[stationIndex,
trainIndex].ToString(twentyFourHourFormat),"Southbound" };

if (this.rdBtnNorthSouth.Checked && checkBoxDown.Checked)
{
dataGridView1.Rows.Add(row);
labelNumberofRowsGenerated.Text = "Total number of rows generated: " + eventNumber.ToString();
}
eventNumber++;
}

}//end of station loop
}//end of train loop
}//end of NS route schedule
//-----------------------------------------------------------------------------------------
else
{
//Clear dataGridView before excuting the schedule
if (this.dataGridView1.DataSource != null)
{
this.dataGridView1.DataSource = null;
}
else
{

{
this.dataGridView1.Rows.Clear();
}
string myMessage = "Please set the start & end of operation times and " +
"choose specific route and direction(s)" +
"of the route you want to make a train timetable!\n\n\n" +
"-----------------------------------------------------------------\n" + "        " +
"Thank you for using CBTSO scheduling Tool.\n" +
"-----------------------------------------------------------------";
```

```
MessageBox.Show(myMessage,
"Warning!",
MessageBoxButtons.OKCancel,
MessageBoxIcon.Asterisk);
//this.BackColor = Color.Red;
}

}

}

private void splitContainer2_Panel2_Paint(object sender, PaintEventArgs e)
{

}

private void helpManualToolStripMenuItem_Click(object sender, EventArgs e)
{
}
private void exportToolStripMenuItem_Click(object sender, EventArgs e)
{
ExportAsExcelFile();
}
private void saveAsToolStripMenuItem_Click(object sender, EventArgs e)
  {
     ExportAsExcelFile();
  }
 }
}
```



S/N	From	To	Distance(m)	Avg Speed(km/hr)	Runing Time	Total Runing Time upto current station(min)	Rounded Runing Time(min)	Total Time upto current station(min)	Total Runing Time upto current station with dwell time included(min)
1	Reveres track	EW01	182	20	0.55	0.55	1	1	1.55
2	EW01	EW02	1260	20	3.78	4.33	4	5	6.33
3	EW02	EW03	1092	20	3.28	7.6	3	8	10.6
4	EW03	EW04	863	20	2.59	10.19	3	11	14.19
5	EW04	EW05	860	20	2.58	12.77	3	14	17.77
6	EW05	EW06	725	20	2.17	14.95	2	16	20.95
7	EW06	EW07	970	20	2.91	17.86	3	19	24.86
8	EW07	EW08	1083	20	3.25	21.1	3	22	29.1
9	EW08	EW09	804	20	2.41	23.52	2	24	32.52
10	EW09	EW10	802	20	2.41	25.92	2	26	35.92
11	EW10	EW11	746	20	2.24	28.16	2	28	39.16
12	EW11	EW12	770	20	2.31	30.47	2	30	42.47

Figure 7.1 Average traveling time for EW route

Table 7.1 Exported result for average Traveling Time on EW line

S/N	From	To	Distance(m)	Avg Speed(km/hr)	Running Time	Total Running Time up to current station(min)	Rounded Running Time(min)	Total Time up to current station(min)	Total Running Time up to current station with dwell time included(min)

1	Reveres track	EW01	182	20	0.55	0.55	1	1	1.55
2	EW01	EW02	1260	20	3.78	4.33	4	5	6.33
3	EW02	EW03	1092	20	3.28	7.6	3	8	10.6
4	EW03	EW04	863	20	2.59	10.19	3	11	14.19
5	EW04	EW05	860	20	2.58	12.77	3	14	17.77
6	EW05	EW06	725	20	2.17	14.95	2	16	20.95
7	EW06	EW07	970	20	2.91	17.86	3	19	24.86
8	EW07	EW08	1083	20	3.25	21.1	3	22	29.1
9	EW08	EW09	804	20	2.41	23.52	2	24	32.52
10	EW09	EW10	802	20	2.41	25.92	2	26	35.92
11	EW10	EW11	746	20	2.24	28.16	2	28	39.16
12	EW11	EW12	770	20	2.31	30.47	2	30	42.47
13	EW12	EW13	952	20	2.86	33.33	3	33	46.33
14	EW13	EW14	675	20	2.02	35.35	2	35	49.35
15	EW14	EW15	583	20	1.75	37.1	2	37	52.1
16	EW15	EW16	650	20	1.95	39.05	2	39	55.05
17	EW16	EW17	435	20	1.3	40.36	1	40	57.36
18	EW17	EW18	570	20	1.71	42.07	2	42	60.07
19	EW18	EW19	688	20	2.06	44.13	2	44	63.13
20	EW19	EW20	735	20	2.2	46.33	2	46	66.34
21	EW20	EW21	725	20	2.17	48.51	2	48	69.51
22	EW21	EW22	776	20	2.33	50.84	2	50	72.84
23	EW22	Buffer stop	908	20	2.72	53.56	3	53	76.56
24	Buffer Stop	EW22	908	20	2.72	56.29	3	56	80.29
25	EW22	EW21	776	20	2.33	58.61	2	58	83.61
26	EW21	EW20	725	20	2.17	60.79	2	60	86.79
27	EW20	EW19	735	20	2.2	62.99	2	62	89.99
28	EW19	EW18	688	20	2.06	65.06	2	64	93.06
29	EW18	EW17	570	20	1.71	66.77	2	66	95.77
30	EW17	EW16	435	20	1.3	68.07	1	67	98.07
31	EW16	EW15	650	20	1.95	70.02	2	69	101.02
32	EW15	EW14	583	20	1.75	71.77	2	71	103.77
33	EW14	EW13	675	20	2.02	73.8	2	73	106.8
34	EW13	EW12	952	20	2.86	76.65	3	76	110.65
35	EW12	EW11	770	20	2.31	78.96	2	78	113.96
36	EW11	EW10	746	20	2.24	81.2	2	80	117.2
37	EW10	EW09	802	20	2.41	83.61	2	82	120.61
38	EW09	EW08	804	20	2.41	86.02	2	84	124.02
39	EW08	EW07	1083	20	3.25	89.27	3	87	128.27
40	EW07	EW06	970	20	2.91	92.18	3	90	132.18
41	EW06	EW05	725	20	2.17	94.35	2	92	135.35
42	EW05	EW04	860	20	2.58	96.93	3	95	138.93
43	EW04	EW03	863	20	2.59	99.52	3	98	142.52

44	EW03	EW02	1092	20	3.28	102.8	3	101	146.8
45	EW02	EW01	1260	20	3.78	106.58	4	105	151.58
46	EW01	Revers e track	182	20	0.55	107.12	1	106	153.12

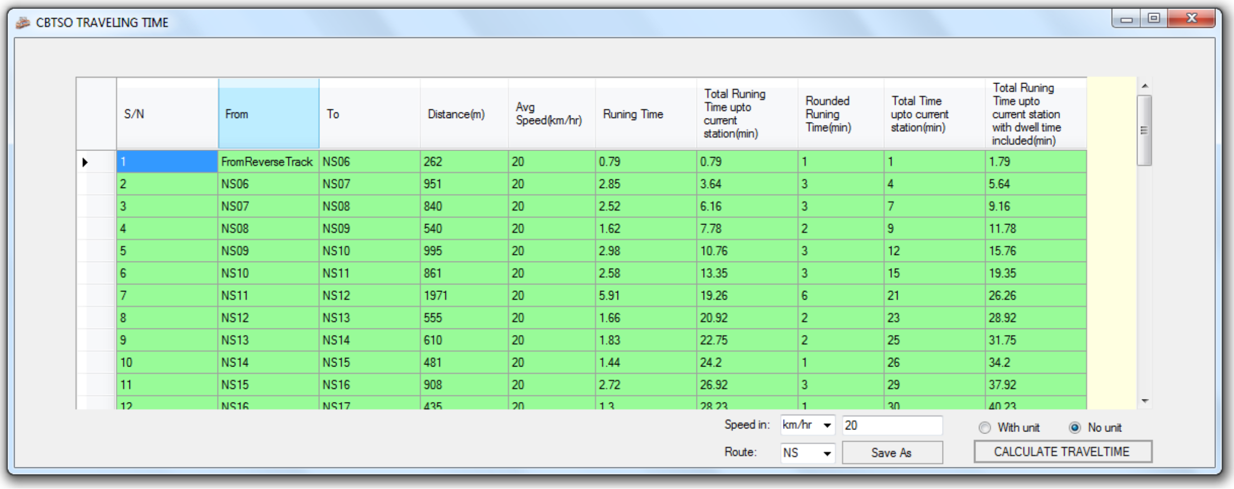


Figure 7.2 Average traveling time results for NS line

Table 7.2 Exported result for average traveling time on NS line

S/N	From	To	Distance(m)	Avg Speed(km/hr)	Running Time	Total Running Time up to current station(min)	Rounded Running Time(min)	Total Time up to current station(min)	Total Running Time up to current station with dwell time included(min)
1	From Revers e Track	NS06	262	20	0.79	0.79	1	1	1.79
2	NS06	NS07	951	20	2.85	3.64	3	4	5.64
3	NS07	NS08	840	20	2.52	6.16	3	7	9.16
4	NS08	NS09	540	20	1.62	7.78	2	9	11.78
5	NS09	NS10	995	20	2.98	10.76	3	12	15.76
6	NS10	NS11	861	20	2.58	13.35	3	15	19.35
7	NS11	NS12	1971	20	5.91	19.26	6	21	26.26
8	NS12	NS13	555	20	1.66	20.92	2	23	28.92
9	NS13	NS14	610	20	1.83	22.75	2	25	31.75
10	NS14	NS15	481	20	1.44	24.20	1	26	34.20
11	NS15	NS16	908	20	2.72	26.92	3	29	37.92
12	NS16	NS17	435	20	1.30	28.23	1	30	40.23
13	NS17	NS18	570	20	1.71	29.94	2	32	42.94
14	NS18	NS19	688	20	2.06	32.00	2	34	46.00
15	NS19	NS20	735	20	2.20	34.21	2	36	49.21
16	NS20	NS21	591	20	1.77	35.98	2	38	51.98
17	NS21	NS22	739	20	2.22	38.20	2	40	55.20

18	NS22	NS23	866	20	2.60	40.79	3	43	58.79
19	NS23	NS24	612	20	1.84	42.63	2	45	61.63
20	NS24	NS25	608	20	1.82	44.45	2	47	64.45
21	NS25	NS26	945	20	2.84	47.29	3	50	68.29
22	NS26	NS27	741	20	2.22	49.51	2	52	71.51
23	NS27	Revers e Track	184	20	0.55	50.06	1	53	73.06
24	Revers e Track	NS27	184	20	0.55	50.62	1	54	74.62
25	NS27	NS26	741	20	2.22	52.84	2	56	77.84
26	NS26	NS25	945	20	2.84	55.67	3	59	81.67
27	NS25	NS24	608	20	1.82	57.50	2	61	84.50
28	NS24	NS23	612	20	1.84	59.33	2	63	87.33
29	NS23	NS22	866	20	2.60	61.93	3	66	90.93
30	NS22	NS21	739	20	2.22	64.15	2	68	94.15
31	NS21	NS20	591	20	1.77	65.92	2	70	96.92
32	NS20	NS19	735	20	2.20	68.13	2	72	100.13
33	NS19	NS18	688	20	2.06	70.19	2	74	103.19
34	NS18	NS17	570	20	1.71	71.90	2	76	105.90
35	NS17	NS16	435	20	1.30	73.21	1	77	108.21
36	NS16	NS15	908	20	2.72	75.93	3	80	111.93
37	NS15	NS14	481	20	1.44	77.37	1	81	114.37
38	NS14	NS13	610	20	1.83	79.20	2	83	117.20
39	NS13	NS12	555	20	1.66	80.87	2	85	119.87
40	NS12	NS11	1971	20	5.91	86.78	6	91	126.78
41	NS11	NS10	861	20	2.58	89.36	3	94	130.36
42	NS10	NS09	995	20	2.98	92.35	3	97	134.35
43	NS09	NS08	540	20	1.62	93.97	2	99	136.97
44	NS08	NS07	840	20	2.52	96.49	3	102	140.49
45	NS07	NS06	951	20	2.85	99.34	3	105	144.34
46	NS06	From Revers e Track	262	20	0.79	100.13	1	106	146.13

Table 7.3: Sample timetable truncated output for EW route

Addis Ababa LRT Passenger trains Daily Timetable Date:2015-Apr-02 02:44 NS route					
Event Number	**Train ID**	**Station ID**	**Arrival Time**	**Departure Time**	**Route Direction**
0001	AALRT-T-01	Station EW-01	05:00	05:01	Westbound
0002	AALRT-T-01	Station EW-02	05:04	05:05	Westbound
0003	AALRT-T-01	Station EW-03	05:08	05:09	Westbound
0004	AALRT-T-01	Station EW-04	05:12	05:13	Westbound
0005	AALRT-T-01	Station EW-05	05:16	05:17	Westbound
0006	AALRT-T-01	Station EW-06	05:20	05:21	Westbound
0007	AALRT-T-01	Station EW-07	05:24	05:25	Westbound
0008	AALRT-T-01	Station EW-08	05:28	05:29	Westbound
0009	AALRT-T-01	Station EW-09	05:32	05:33	Westbound

0010	AALRT-T-01	Station EW-10	05:36	05:37	Westbound
0011	AALRT-T-01	Station EW-11	05:40	05:41	Westbound
0012	AALRT-T-01	Station EW-12	05:44	05:45	Westbound
0013	AALRT-T-01	Station EW-13	05:48	05:49	Westbound
0014	AALRT-T-01	Station EW-14	05:52	05:53	Westbound
0015	AALRT-T-01	Station EW-15	05:56	05:57	Westbound
0016	AALRT-T-01	Station EW-16	06:00	06:01	Westbound
0017	AALRT-T-01	Station EW-17	06:04	06:05	Westbound
0018	AALRT-T-01	Station EW-18	06:08	06:09	Westbound
0019	AALRT-T-01	Station EW-19	06:12	06:13	Westbound
0020	AALRT-T-01	Station EW-20	06:16	06:17	Westbound
0021	AALRT-T-01	Station EW-21	06:20	06:21	Westbound
0022	AALRT-T-01	Station EW-22	06:24	06:25	Westbound
0023	AALRT-T-01	Station EW-21	06:28	06:29	Eastbound
0024	AALRT-T-01	Station EW-20	06:32	06:33	Eastbound
0025	AALRT-T-01	Station EW-19	06:36	06:37	Eastbound
0026	AALRT-T-01	Station EW-18	06:40	06:41	Eastbound
0027	AALRT-T-01	Station EW-17	06:44	06:45	Eastbound
0028	AALRT-T-01	Station EW-16	06:48	06:49	Eastbound
0029	AALRT-T-01	Station EW-15	06:52	06:53	Eastbound
0030	AALRT-T-01	Station EW-14	06:56	06:57	Eastbound
0031	AALRT-T-01	Station EW-13	07:00	07:01	Eastbound
0032	AALRT-T-01	Station EW-12	07:04	07:05	Eastbound
0033	AALRT-T-01	Station EW-11	07:08	07:09	Eastbound
0034	AALRT-T-01	Station EW-10	07:12	07:13	Eastbound
0035	AALRT-T-01	Station EW-09	07:16	07:17	Eastbound
0036	AALRT-T-01	Station EW-08	07:20	07:21	Eastbound
0037	AALRT-T-01	Station EW-07	07:24	07:25	Eastbound
0038	AALRT-T-01	Station EW-06	07:28	07:29	Eastbound
0039	AALRT-T-01	Station EW-05	07:32	07:33	Eastbound
0040	AALRT-T-01	Station EW-04	07:36	07:37	Eastbound
0041	AALRT-T-01	Station EW-03	07:40	07:41	Eastbound
0042	AALRT-T-01	Station EW-02	07:44	07:45	Eastbound
0043	AALRT-T-01	Station EW-01	07:48	07:49	Eastbound
0044	AALRT-T-01	Station EW-00	07:52	07:53	Eastbound
0045	AALRT-T-02	Station EW-01	05:06	05:07	Westbound
0046	AALRT-T-02	Station EW-02	05:10	05:11	Westbound
0047	AALRT-T-02	Station EW-03	05:14	05:15	Westbound
0048	AALRT-T-02	Station EW-04	05:18	05:19	Westbound
0049	AALRT-T-02	Station EW-05	05:22	05:23	Westbound
0050	AALRT-T-02	Station EW-06	05:26	05:27	Westbound
0051	AALRT-T-02	Station EW-07	05:30	05:31	Westbound

0052	AALRT-T-02	Station EW-08	05:34	05:35	Westbound
0053	AALRT-T-02	Station EW-09	05:38	05:39	Westbound
0054	AALRT-T-02	Station EW-10	05:42	05:43	Westbound
0055	AALRT-T-02	Station EW-11	05:46	05:47	Westbound
0056	AALRT-T-02	Station EW-12	05:50	05:51	Westbound
0057	AALRT-T-02	Station EW-13	05:54	05:55	Westbound
0058	AALRT-T-02	Station EW-14	05:58	05:59	Westbound
0059	AALRT-T-02	Station EW-15	06:02	06:03	Westbound
0060	AALRT-T-02	Station EW-16	06:06	06:07	Westbound
0061	AALRT-T-02	Station EW-17	06:10	06:11	Westbound
0062	AALRT-T-02	Station EW-18	06:14	06:15	Westbound
0063	AALRT-T-02	Station EW-19	06:18	06:19	Westbound
0064	AALRT-T-02	Station EW-20	06:22	06:23	Westbound
0065	AALRT-T-02	Station EW-21	06:26	06:27	Westbound
0066	AALRT-T-02	Station EW-22	06:30	06:31	Westbound
0067	AALRT-T-02	Station EW-21	06:34	06:35	Eastbound
0068	AALRT-T-02	Station EW-20	06:38	06:39	Eastbound
0069	AALRT-T-02	Station EW-19	06:42	06:43	Eastbound
0070	AALRT-T-02	Station EW-18	06:46	06:47	Eastbound
0071	AALRT-T-02	Station EW-17	06:50	06:51	Eastbound
0072	AALRT-T-02	Station EW-16	06:54	06:55	Eastbound
0073	AALRT-T-02	Station EW-15	06:58	06:59	Eastbound
0074	AALRT-T-02	Station EW-14	07:02	07:03	Eastbound
0075	AALRT-T-02	Station EW-13	07:06	07:07	Eastbound
0076	AALRT-T-02	Station EW-12	07:10	07:11	Eastbound
0077	AALRT-T-02	Station EW-11	07:14	07:15	Eastbound
0078	AALRT-T-02	Station EW-10	07:18	07:19	Eastbound
0079	AALRT-T-02	Station EW-09	07:22	07:23	Eastbound
0080	AALRT-T-02	Station EW-08	07:26	07:27	Eastbound
0081	AALRT-T-02	Station EW-07	07:30	07:31	Eastbound
0082	AALRT-T-02	Station EW-06	07:34	07:35	Eastbound
0083	AALRT-T-02	Station EW-05	07:38	07:39	Eastbound
0084	AALRT-T-02	Station EW-04	07:42	07:43	Eastbound
0085	AALRT-T-02	Station EW-03	07:46	07:47	Eastbound
0086	AALRT-T-02	Station EW-02	07:50	07:51	Eastbound
0087	AALRT-T-02	Station EW-01	07:54	07:55	Eastbound
0088	AALRT-T-02	Station EW-00	07:58	07:59	Eastbound
0089	AALRT-T-03	Station EW-01	05:12	05:13	Westbound
0090	AALRT-T-03	Station EW-02	05:16	05:17	Westbound
0091	AALRT-T-03	Station EW-03	05:20	05:21	Westbound
0092	AALRT-T-03	Station EW-04	05:24	05:25	Westbound
0093	AALRT-T-03	Station EW-05	05:28	05:29	Westbound

0094	AALRT-T-03	Station EW-06	05:32	05:33	Westbound
0095	AALRT-T-03	Station EW-07	05:36	05:37	Westbound
0096	AALRT-T-03	Station EW-08	05:40	05:41	Westbound
0097	AALRT-T-03	Station EW-09	05:44	05:45	Westbound
0098	AALRT-T-03	Station EW-10	05:48	05:49	Westbound
0099	AALRT-T-03	Station EW-11	05:52	05:53	Westbound
0100	AALRT-T-03	Station EW-12	05:56	05:57	Westbound
0101	AALRT-T-03	Station EW-13	06:00	06:01	Westbound
0102	AALRT-T-03	Station EW-14	06:04	06:05	Westbound
0103	AALRT-T-03	Station EW-15	06:08	06:09	Westbound
0104	AALRT-T-03	Station EW-16	06:12	06:13	Westbound
0105	AALRT-T-03	Station EW-17	06:16	06:17	Westbound
0106	AALRT-T-03	Station EW-18	06:20	06:21	Westbound
0107	AALRT-T-03	Station EW-19	06:24	06:25	Westbound
0108	AALRT-T-03	Station EW-20	06:28	06:29	Westbound
0109	AALRT-T-03	Station EW-21	06:32	06:33	Westbound
0110	AALRT-T-03	Station EW-22	06:36	06:37	Westbound
0111	AALRT-T-03	Station EW-21	06:40	06:41	Eastbound
0112	AALRT-T-03	Station EW-20	06:44	06:45	Eastbound
0113	AALRT-T-03	Station EW-19	06:48	06:49	Eastbound
0114	AALRT-T-03	Station EW-18	06:52	06:53	Eastbound
0115	AALRT-T-03	Station EW-17	06:56	06:57	Eastbound
0116	AALRT-T-03	Station EW-16	07:00	07:01	Eastbound
0117	AALRT-T-03	Station EW-15	07:04	07:05	Eastbound
0118	AALRT-T-03	Station EW-14	07:08	07:09	Eastbound
0119	AALRT-T-03	Station EW-13	07:12	07:13	Eastbound
0120	AALRT-T-03	Station EW-12	07:16	07:17	Eastbound
0121	AALRT-T-03	Station EW-11	07:20	07:21	Eastbound
0122	AALRT-T-03	Station EW-10	07:24	07:25	Eastbound
0123	AALRT-T-03	Station EW-09	07:28	07:29	Eastbound
0124	AALRT-T-03	Station EW-08	07:32	07:33	Eastbound
0125	AALRT-T-03	Station EW-07	07:36	07:37	Eastbound
0126	AALRT-T-03	Station EW-06	07:40	07:41	Eastbound
0127	AALRT-T-03	Station EW-05	07:44	07:45	Eastbound
0128	AALRT-T-03	Station EW-04	07:48	07:49	Eastbound
0129	AALRT-T-03	Station EW-03	07:52	07:53	Eastbound
0130	AALRT-T-03	Station EW-02	07:56	07:57	Eastbound
0131	AALRT-T-03	Station EW-01	08:00	08:01	Eastbound
0132	AALRT-T-03	Station EW-00	08:04	08:05	Eastbound
0133	AALRT-T-04	Station EW-01	05:18	05:19	Westbound
0134	AALRT-T-04	Station EW-02	05:22	05:23	Westbound
0135	AALRT-T-04	Station EW-03	05:26	05:27	Westbound

0136	AALRT-T-04	Station EW-04	05:30	05:31	Westbound
0137	AALRT-T-04	Station EW-05	05:34	05:35	Westbound
0138	AALRT-T-04	Station EW-06	05:38	05:39	Westbound
0139	AALRT-T-04	Station EW-07	05:42	05:43	Westbound
0140	AALRT-T-04	Station EW-08	05:46	05:47	Westbound
0141	AALRT-T-04	Station EW-09	05:50	05:51	Westbound
0142	AALRT-T-04	Station EW-10	05:54	05:55	Westbound
0143	AALRT-T-04	Station EW-11	05:58	05:59	Westbound
0144	AALRT-T-04	Station EW-12	06:02	06:03	Westbound
0145	AALRT-T-04	Station EW-13	06:06	06:07	Westbound
0146	AALRT-T-04	Station EW-14	06:10	06:11	Westbound
0147	AALRT-T-04	Station EW-15	06:14	06:15	Westbound
0148	AALRT-T-04	Station EW-16	06:18	06:19	Westbound
0149	AALRT-T-04	Station EW-17	06:22	06:23	Westbound
0150	AALRT-T-04	Station EW-18	06:26	06:27	Westbound
0151	AALRT-T-04	Station EW-19	06:30	06:31	Westbound
0152	AALRT-T-04	Station EW-20	06:34	06:35	Westbound
0153	AALRT-T-04	Station EW-21	06:38	06:39	Westbound
0154	AALRT-T-04	Station EW-22	06:42	06:43	Westbound
0155	AALRT-T-04	Station EW-21	06:46	06:47	Eastbound
0156	AALRT-T-04	Station EW-20	06:50	06:51	Eastbound
0157	AALRT-T-04	Station EW-19	06:54	06:55	Eastbound
0158	AALRT-T-04	Station EW-18	06:58	06:59	Eastbound
0159	AALRT-T-04	Station EW-17	07:02	07:03	Eastbound
0160	AALRT-T-04	Station EW-16	07:06	07:07	Eastbound
0161	AALRT-T-04	Station EW-15	07:10	07:11	Eastbound
0162	AALRT-T-04	Station EW-14	07:14	07:15	Eastbound
0163	AALRT-T-04	Station EW-13	07:18	07:19	Eastbound
0164	AALRT-T-04	Station EW-12	07:22	07:23	Eastbound
0165	AALRT-T-04	Station EW-11	07:26	07:27	Eastbound
0166	AALRT-T-04	Station EW-10	07:30	07:31	Eastbound
0167	AALRT-T-04	Station EW-09	07:34	07:35	Eastbound
0168	AALRT-T-04	Station EW-08	07:38	07:39	Eastbound
0169	AALRT-T-04	Station EW-07	07:42	07:43	Eastbound
0170	AALRT-T-04	Station EW-06	07:46	07:47	Eastbound
0171	AALRT-T-04	Station EW-05	07:50	07:51	Eastbound
0172	AALRT-T-04	Station EW-04	07:54	07:55	Eastbound
0173	AALRT-T-04	Station EW-03	07:58	07:59	Eastbound
0174	AALRT-T-04	Station EW-02	08:02	08:03	Eastbound
0175	AALRT-T-04	Station EW-01	08:06	08:07	Eastbound
0176	AALRT-T-04	Station EW-00	08:10	08:11	Eastbound
0177	AALRT-T-05	Station EW-01	05:24	05:25	Westbound

0178	AALRT-T-05	Station EW-02	05:28	05:29	Westbound
0179	AALRT-T-05	Station EW-03	05:32	05:33	Westbound
0180	AALRT-T-05	Station EW-04	05:36	05:37	Westbound
0181	AALRT-T-05	Station EW-05	05:40	05:41	Westbound
0182	AALRT-T-05	Station EW-06	05:44	05:45	Westbound
0183	AALRT-T-05	Station EW-07	05:48	05:49	Westbound
0184	AALRT-T-05	Station EW-08	05:52	05:53	Westbound
0185	AALRT-T-05	Station EW-09	05:56	05:57	Westbound
0186	AALRT-T-05	Station EW-10	06:00	06:01	Westbound
0187	AALRT-T-05	Station EW-11	06:04	06:05	Westbound
0188	AALRT-T-05	Station EW-12	06:08	06:09	Westbound
0189	AALRT-T-05	Station EW-13	06:12	06:13	Westbound
0190	AALRT-T-05	Station EW-14	06:16	06:17	Westbound
0191	AALRT-T-05	Station EW-15	06:20	06:21	Westbound
0192	AALRT-T-05	Station EW-16	06:24	06:25	Westbound
0193	AALRT-T-05	Station EW-17	06:28	06:29	Westbound
0194	AALRT-T-05	Station EW-18	06:32	06:33	Westbound
0195	AALRT-T-05	Station EW-19	06:36	06:37	Westbound
0196	AALRT-T-05	Station EW-20	06:40	06:41	Westbound
0197	AALRT-T-05	Station EW-21	06:44	06:45	Westbound
0198	AALRT-T-05	Station EW-22	06:48	06:49	Westbound
0199	AALRT-T-05	Station EW-21	06:52	06:53	Eastbound
0200	AALRT-T-05	Station EW-20	06:56	06:57	Eastbound
0201	AALRT-T-05	Station EW-19	07:00	07:01	Eastbound
0202	AALRT-T-05	Station EW-18	07:04	07:05	Eastbound
0203	AALRT-T-05	Station EW-17	07:08	07:09	Eastbound
0204	AALRT-T-05	Station EW-16	07:12	07:13	Eastbound
0205	AALRT-T-05	Station EW-15	07:16	07:17	Eastbound
…	…	…	…	…	…
…	…	…	…	…	…
7164	AALRT-T-03	Station EW-08	23:32	23:33	Eastbound
7165	AALRT-T-03	Station EW-07	23:36	23:37	Eastbound
7166	AALRT-T-03	Station EW-06	23:40	23:41	Eastbound
7167	AALRT-T-03	Station EW-05	23:44	23:45	Eastbound
7168	AALRT-T-03	Station EW-04	23:48	23:49	Eastbound
7169	AALRT-T-03	Station EW-03	23:52	23:53	Eastbound
7170	AALRT-T-03	Station EW-02	23:56	23:57	Eastbound
7171	AALRT-T-03	Station EW-01	00:00	00:01	Eastbound
7172	AALRT-T-03	Station EW-00	00:04	00:05	Eastbound
7173	AALRT-T-04	Station EW-01	21:18	21:19	Westbound
7174	AALRT-T-04	Station EW-02	21:22	21:23	Westbound
7175	AALRT-T-04	Station EW-03	21:26	21:27	Westbound

7176	AALRT-T-04	Station EW-04	21:30	21:31	Westbound
7177	AALRT-T-04	Station EW-05	21:34	21:35	Westbound
7178	AALRT-T-04	Station EW-06	21:38	21:39	Westbound
7179	AALRT-T-04	Station EW-07	21:42	21:43	Westbound
7180	AALRT-T-04	Station EW-08	21:46	21:47	Westbound
7181	AALRT-T-04	Station EW-09	21:50	21:51	Westbound
7182	AALRT-T-04	Station EW-10	21:54	21:55	Westbound
7183	AALRT-T-04	Station EW-11	21:58	21:59	Westbound
7184	AALRT-T-04	Station EW-12	22:02	22:03	Westbound
7185	AALRT-T-04	Station EW-13	22:06	22:07	Westbound
7186	AALRT-T-04	Station EW-14	22:10	22:11	Westbound
7187	AALRT-T-04	Station EW-15	22:14	22:15	Westbound
7188	AALRT-T-04	Station EW-16	22:18	22:19	Westbound
7189	AALRT-T-04	Station EW-17	22:22	22:23	Westbound
7190	AALRT-T-04	Station EW-18	22:26	22:27	Westbound
7191	AALRT-T-04	Station EW-19	22:30	22:31	Westbound
7192	AALRT-T-04	Station EW-20	22:34	22:35	Westbound
7193	AALRT-T-04	Station EW-21	22:38	22:39	Westbound
7194	AALRT-T-04	Station EW-22	22:42	22:43	Westbound
7195	AALRT-T-04	Station EW-21	22:46	22:47	Eastbound
7196	AALRT-T-04	Station EW-20	22:50	22:51	Eastbound
7197	AALRT-T-04	Station EW-19	22:54	22:55	Eastbound
7198	AALRT-T-04	Station EW-18	22:58	22:59	Eastbound

Table 7.4: Sample timetable truncated output for NS route

Addis Ababa LRT Passenger trains Daily Timetable Date:2015-Apr-02 02:25 NS route					
Event Number	**Train ID**	**Station ID**	**Arrival Time**	**Departure Time**	**Route Direction**
0001	AALRT-T-21	Station NS-06	05:03	05:04	Northbound
0002	AALRT-T-21	Station NS-07	05:07	05:08	Northbound
0003	AALRT-T-21	Station NS-08	05:11	05:12	Northbound
0004	AALRT-T-21	Station NS-09	05:15	05:16	Northbound
0005	AALRT-T-21	Station NS-10	05:19	05:20	Northbound
0006	AALRT-T-21	Station NS-11	05:23	05:24	Northbound
0007	AALRT-T-21	Station NS-12	05:27	05:28	Northbound
0008	AALRT-T-21	Station NS-13	05:31	05:32	Northbound
0009	AALRT-T-21	Station NS-14	05:35	05:36	Northbound
0010	AALRT-T-21	Station NS-15	05:39	05:40	Northbound
0011	AALRT-T-21	Station NS-16	05:43	05:44	Northbound
0012	AALRT-T-21	Station NS-17	05:47	05:48	Northbound
0013	AALRT-T-21	Station NS-18	05:51	05:52	Northbound
0014	AALRT-T-21	Station NS-19	05:55	05:56	Northbound
0015	AALRT-T-21	Station NS-20	05:59	06:00	Northbound

0016	AALRT-T-21	Station NS-21	06:03	06:04	Northbound
0017	AALRT-T-21	Station NS-22	06:07	06:08	Northbound
0018	AALRT-T-21	Station NS-23	06:11	06:12	Northbound
0019	AALRT-T-21	Station NS-24	06:15	06:16	Northbound
0020	AALRT-T-21	Station NS-25	06:19	06:20	Northbound
0021	AALRT-T-21	Station NS-26	06:23	06:24	Northbound
0022	AALRT-T-21	Station NS-27	06:27	06:28	Northbound
0023	AALRT-T-21	Station NS-26	06:31	06:32	Southbound
0024	AALRT-T-21	Station NS-25	06:35	06:36	Southbound
0025	AALRT-T-21	Station NS-24	06:39	06:40	Southbound
0026	AALRT-T-21	Station NS-23	06:43	06:44	Southbound
0027	AALRT-T-21	Station NS-22	06:47	06:48	Southbound
0028	AALRT-T-21	Station NS-21	06:51	06:52	Southbound
0029	AALRT-T-21	Station NS-20	06:55	06:56	Southbound
0030	AALRT-T-21	Station NS-19	06:59	07:00	Southbound
0031	AALRT-T-21	Station NS-18	07:03	07:04	Southbound
0032	AALRT-T-21	Station NS-17	07:07	07:08	Southbound
0033	AALRT-T-21	Station NS-16	07:11	07:12	Southbound
0034	AALRT-T-21	Station NS-15	07:15	07:16	Southbound
0035	AALRT-T-21	Station NS-14	07:19	07:20	Southbound
0036	AALRT-T-21	Station NS-13	07:23	07:24	Southbound
0037	AALRT-T-21	Station NS-12	07:27	07:28	Southbound
0038	AALRT-T-21	Station NS-11	07:31	07:32	Southbound
0039	AALRT-T-21	Station NS-10	07:35	07:36	Southbound
0040	AALRT-T-21	Station NS-09	07:39	07:40	Southbound
0041	AALRT-T-21	Station NS-08	07:43	07:44	Southbound
0042	AALRT-T-21	Station NS-07	07:47	07:48	Southbound
0043	AALRT-T-21	Station NS-06	07:51	07:52	Southbound
0044	AALRT-T-21	Station NS-05	07:55	07:56	Southbound
0045	AALRT-T-22	Station NS-06	05:06	05:07	Northbound
0046	AALRT-T-22	Station NS-07	05:10	05:11	Northbound
0047	AALRT-T-22	Station NS-08	05:14	05:15	Northbound
0048	AALRT-T-22	Station NS-09	05:18	05:19	Northbound
0049	AALRT-T-22	Station NS-10	05:22	05:23	Northbound
0050	AALRT-T-22	Station NS-11	05:26	05:27	Northbound
0051	AALRT-T-22	Station NS-12	05:30	05:31	Northbound
0052	AALRT-T-22	Station NS-13	05:34	05:35	Northbound
0053	AALRT-T-22	Station NS-14	05:38	05:39	Northbound
0054	AALRT-T-22	Station NS-15	05:42	05:43	Northbound
0055	AALRT-T-22	Station NS-16	05:46	05:47	Northbound
0056	AALRT-T-22	Station NS-17	05:50	05:51	Northbound
0057	AALRT-T-22	Station NS-18	05:54	05:55	Northbound
0058	AALRT-T-22	Station NS-19	05:58	05:59	Northbound
0059	AALRT-T-22	Station NS-20	06:02	06:03	Northbound
0060	AALRT-T-22	Station NS-21	06:06	06:07	Northbound
0061	AALRT-T-22	Station NS-22	06:10	06:11	Northbound
0062	AALRT-T-22	Station NS-23	06:14	06:15	Northbound
0063	AALRT-T-22	Station NS-24	06:18	06:19	Northbound
0064	AALRT-T-22	Station NS-25	06:22	06:23	Northbound
0065	AALRT-T-22	Station NS-26	06:26	06:27	Northbound
0066	AALRT-T-22	Station NS-27	06:30	06:31	Northbound

0067	AALRT-T-22	Station NS-26	06:34	06:35	Southbound
0068	AALRT-T-22	Station NS-25	06:38	06:39	Southbound
0069	AALRT-T-22	Station NS-24	06:42	06:43	Southbound
0070	AALRT-T-22	Station NS-23	06:46	06:47	Southbound
0071	AALRT-T-22	Station NS-22	06:50	06:51	Southbound
0072	AALRT-T-22	Station NS-21	06:54	06:55	Southbound
0073	AALRT-T-22	Station NS-20	06:58	06:59	Southbound
0074	AALRT-T-22	Station NS-19	07:02	07:03	Southbound
0075	AALRT-T-22	Station NS-18	07:06	07:07	Southbound
0076	AALRT-T-22	Station NS-17	07:10	07:11	Southbound
0077	AALRT-T-22	Station NS-16	07:14	07:15	Southbound
0078	AALRT-T-22	Station NS-15	07:18	07:19	Southbound
0079	AALRT-T-22	Station NS-14	07:22	07:23	Southbound
0080	AALRT-T-22	Station NS-13	07:26	07:27	Southbound
0081	AALRT-T-22	Station NS-12	07:30	07:31	Southbound
0082	AALRT-T-22	Station NS-11	07:34	07:35	Southbound
0083	AALRT-T-22	Station NS-10	07:38	07:39	Southbound
0084	AALRT-T-22	Station NS-09	07:42	07:43	Southbound
0085	AALRT-T-22	Station NS-08	07:46	07:47	Southbound
0086	AALRT-T-22	Station NS-07	07:50	07:51	Southbound
0087	AALRT-T-22	Station NS-06	07:54	07:55	Southbound
0088	AALRT-T-22	Station NS-05	07:58	07:59	Southbound
0089	AALRT-T-23	Station NS-06	05:12	05:13	Northbound
0090	AALRT-T-23	Station NS-07	05:16	05:17	Northbound
0091	AALRT-T-23	Station NS-08	05:20	05:21	Northbound
0092	AALRT-T-23	Station NS-09	05:24	05:25	Northbound
0093	AALRT-T-23	Station NS-10	05:28	05:29	Northbound
0094	AALRT-T-23	Station NS-11	05:32	05:33	Northbound
0095	AALRT-T-23	Station NS-12	05:36	05:37	Northbound
0096	AALRT-T-23	Station NS-13	05:40	05:41	Northbound
0097	AALRT-T-23	Station NS-14	05:44	05:45	Northbound
0098	AALRT-T-23	Station NS-15	05:48	05:49	Northbound
0099	AALRT-T-23	Station NS-16	05:52	05:53	Northbound
0100	AALRT-T-23	Station NS-17	05:56	05:57	Northbound
0101	AALRT-T-23	Station NS-18	06:00	06:01	Northbound
0102	AALRT-T-23	Station NS-19	06:04	06:05	Northbound
0103	AALRT-T-23	Station NS-20	06:08	06:09	Northbound
0104	AALRT-T-23	Station NS-21	06:12	06:13	Northbound
0105	AALRT-T-23	Station NS-22	06:16	06:17	Northbound
0106	AALRT-T-23	Station NS-23	06:20	06:21	Northbound
0107	AALRT-T-23	Station NS-24	06:24	06:25	Northbound
0108	AALRT-T-23	Station NS-25	06:28	06:29	Northbound
0109	AALRT-T-23	Station NS-26	06:32	06:33	Northbound
0110	AALRT-T-23	Station NS-27	06:36	06:37	Northbound
0111	AALRT-T-23	Station NS-26	06:40	06:41	Southbound
0112	AALRT-T-23	Station NS-25	06:44	06:45	Southbound
0113	AALRT-T-23	Station NS-24	06:48	06:49	Southbound
0114	AALRT-T-23	Station NS-23	06:52	06:53	Southbound
0115	AALRT-T-23	Station NS-22	06:56	06:57	Southbound
0116	AALRT-T-23	Station NS-21	07:00	07:01	Southbound
0117	AALRT-T-23	Station NS-20	07:04	07:05	Southbound

0118	AALRT-T-23	Station NS-19	07:08	07:09	Southbound
0119	AALRT-T-23	Station NS-18	07:12	07:13	Southbound
0120	AALRT-T-23	Station NS-17	07:16	07:17	Southbound
0121	AALRT-T-23	Station NS-16	07:20	07:21	Southbound
0122	AALRT-T-23	Station NS-15	07:24	07:25	Southbound
0123	AALRT-T-23	Station NS-14	07:28	07:29	Southbound
0124	AALRT-T-23	Station NS-13	07:32	07:33	Southbound
0125	AALRT-T-23	Station NS-12	07:36	07:37	Southbound
0126	AALRT-T-23	Station NS-11	07:40	07:41	Southbound
0127	AALRT-T-23	Station NS-10	07:44	07:45	Southbound
0128	AALRT-T-23	Station NS-09	07:48	07:49	Southbound
0129	AALRT-T-23	Station NS-08	07:52	07:53	Southbound
0130	AALRT-T-23	Station NS-07	07:56	07:57	Southbound
0131	AALRT-T-23	Station NS-06	08:00	08:01	Southbound
0132	AALRT-T-23	Station NS-05	08:04	08:05	Southbound
0133	AALRT-T-24	Station NS-06	05:18	05:19	Northbound
0134	AALRT-T-24	Station NS-07	05:22	05:23	Northbound
0135	AALRT-T-24	Station NS-08	05:26	05:27	Northbound
0136	AALRT-T-24	Station NS-09	05:30	05:31	Northbound
0137	AALRT-T-24	Station NS-10	05:34	05:35	Northbound
0138	AALRT-T-24	Station NS-11	05:38	05:39	Northbound
0139	AALRT-T-24	Station NS-12	05:42	05:43	Northbound
0140	AALRT-T-24	Station NS-13	05:46	05:47	Northbound
0141	AALRT-T-24	Station NS-14	05:50	05:51	Northbound
0142	AALRT-T-24	Station NS-15	05:54	05:55	Northbound
0143	AALRT-T-24	Station NS-16	05:58	05:59	Northbound
0144	AALRT-T-24	Station NS-17	06:02	06:03	Northbound
0145	AALRT-T-24	Station NS-18	06:06	06:07	Northbound
0146	AALRT-T-24	Station NS-19	06:10	06:11	Northbound
0147	AALRT-T-24	Station NS-20	06:14	06:15	Northbound
0148	AALRT-T-24	Station NS-21	06:18	06:19	Northbound
0149	AALRT-T-24	Station NS-22	06:22	06:23	Northbound
0150	AALRT-T-24	Station NS-23	06:26	06:27	Northbound
0151	AALRT-T-24	Station NS-24	06:30	06:31	Northbound
0152	AALRT-T-24	Station NS-25	06:34	06:35	Northbound
0153	AALRT-T-24	Station NS-26	06:38	06:39	Northbound
0154	AALRT-T-24	Station NS-27	06:42	06:43	Northbound
0155	AALRT-T-24	Station NS-26	06:46	06:47	Southbound
0156	AALRT-T-24	Station NS-25	06:50	06:51	Southbound
0157	AALRT-T-24	Station NS-24	06:54	06:55	Southbound
0158	AALRT-T-24	Station NS-23	06:58	06:59	Southbound
0159	AALRT-T-24	Station NS-22	07:02	07:03	Southbound
0160	AALRT-T-24	Station NS-21	07:06	07:07	Southbound
0161	AALRT-T-24	Station NS-20	07:10	07:11	Southbound
0162	AALRT-T-24	Station NS-19	07:14	07:15	Southbound
0163	AALRT-T-24	Station NS-18	07:18	07:19	Southbound
0164	AALRT-T-24	Station NS-17	07:22	07:23	Southbound
0165	AALRT-T-24	Station NS-16	07:26	07:27	Southbound
0166	AALRT-T-24	Station NS-15	07:30	07:31	Southbound
0167	AALRT-T-24	Station NS-14	07:34	07:35	Southbound
0168	AALRT-T-24	Station NS-13	07:38	07:39	Southbound

0169	AALRT-T-24	Station NS-12	07:42	07:43	Southbound
0170	AALRT-T-24	Station NS-11	07:46	07:47	Southbound
0171	AALRT-T-24	Station NS-10	07:50	07:51	Southbound
0172	AALRT-T-24	Station NS-09	07:54	07:55	Southbound
0173	AALRT-T-24	Station NS-08	07:58	07:59	Southbound
0174	AALRT-T-24	Station NS-07	08:02	08:03	Southbound
0175	AALRT-T-24	Station NS-06	08:06	08:07	Southbound
0176	AALRT-T-24	Station NS-05	08:10	08:11	Southbound
0177	AALRT-T-25	Station NS-06	05:24	05:25	Northbound
0178	AALRT-T-25	Station NS-07	05:28	05:29	Northbound
0179	AALRT-T-25	Station NS-08	05:32	05:33	Northbound
0180	AALRT-T-25	Station NS-09	05:36	05:37	Northbound
…	…	…	…	…	…
…	…	…	…	…	…
7191	AALRT-T-24	Station NS-24	22:30	22:31	Northbound
7192	AALRT-T-24	Station NS-25	22:34	22:35	Northbound
7193	AALRT-T-24	Station NS-26	22:38	22:39	Northbound
7194	AALRT-T-24	Station NS-27	22:42	22:43	Northbound
7195	AALRT-T-24	Station NS-26	22:46	22:47	Southbound
7196	AALRT-T-24	Station NS-25	22:50	22:51	Southbound
7197	AALRT-T-24	Station NS-24	22:54	22:55	Southbound
7198	AALRT-T-24	Station NS-23	22:58	22:59	Southbound