Train Scheduling with Temporal Planning

Md. Amin Kaiser
Md. Nurnoby Mallik
Vocti Chakma

A thesis submitted for the degree of
BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

Department of Computer Science and Engineering
Dhaka University of Engineering & Technology, Gazipur

June 2022

Train Scheduling with Temporal Planning

Md. Amin Kaiser
Student no: 164113
Reg: 9770, Session: 2019-2020

Md. Nurnoby Mallik
Student no: 164066
Reg: 9723, Session: 2019-2020

Vocti Chakma
Student no: 164112
Reg: 9769, Session: 2019-2020

Supervisor: Dr. Fazlul Hasan Siddiqui
Professor

A thesis submitted for the degree of
BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

Department of Computer Science and Engineering
Dhaka University of Engineering & Technology, Gazipur

June 2022

Declaration

Declaration of the Supervisor

I confirm that, to the best of my knowledge, the thesis was carried out and the thesis was prepared under my direct supervision. It represents the original research work of the candidates. I have read this work and, in my opinion, this work is adequate in terms of scope and quality for the purpose of awarding the degree of Bachelor of Science in Computer Science and Engineering. This is certifying that the below statement made by the candidates is correct and true to the best of my knowledge.

.. ..
Dr. Fazlul Hasan Siddiqui
Professor
Department of Computer Science and Engineering
Dhaka University of Engineering & Technology, Gazipur

Candidate's Declaration

We hereby declare that the work which is being presented in the thesis entailed, "Train Scheduling Using Temporal Planning", in partial fulfilment of the award of the degree of Bachelor of Science in Computer Science and Engineering submitted to the department of the Computer Science and Engineering of Dhaka University of Engineering and Technology (DUET), Gazipur, Bangladesh is an authentic record of our own work carried out under the supervision of Dr. Fazlul Hasan Siddiqui during the session 2019-2020.

…………………….
Md. Amin Kaiser
Studen ID: 164113

……………………
Md. Nurnoby Mallik
Student ID: 164066

…………………….
Vocti Chakma
Student ID: 164112

Acknowledgments

The completion of any successful project makes us proud and brings a sense of satisfaction. We would like to express our sincere thanks and gratitude to Almighty Allah that we have been able to successfully complete this thesis and project.

We would like to express our best regards to our supervisor of this work Dr. Fazlul Hasan Siddiqui, Professor, Department of Computer Science and Engineering, DUET, Gazipur, for his excellent guidance and support throughout the whole journey of this work. He introduces us to a modern and interesting field of analysis and provides the required materials, valuable time and professional thoughts. His inspiration, support, constructive criticism, and valuable advice at all stages of our work have made it possible to complete the work. We also like to thank him for deciding to research in 3rd year. Thanks to all of our teachers, and also thanks to all the staff of the CSE department, DUET.

We are extremely happy to acknowledge and express our respect and love to our parents for their constant support and encouragement and last but not least we like to thank our friends and well-wishers for their help and support.

Abstract

In Bangladesh, Railway transportation is considered a comfortable, cheapest, environment-friendly, and efficient mode of transport. The Bangladesh Railway system has a total length of 3,600 kilometres of rail network connecting 44 districts. In 2014, Bangladesh Railway carried 65 million passengers and 2.52 million tons of freight [1]. According to **ADB** (2000), about 20% of the passengers in all transport sectors in Bangladesh travel by train. But train schedule disasters, delays and accidents are the common issues in the railway system. There are some opportunities to optimize the train trip time, route choice, and meeting-crossing between trains.

In this work, we focused on developing automated schedules and plans for the Bangladesh Railway (BR) system in order to optimize the train schedules. We constructed a scheduling model that generates automated schedules also known as plans. We applied these generated automated plans to the **BR** system to minimize the trip time of the train and improve the passenger's service and ease their suffering. In our strategy, to simultaneously execute multiple actions parallelly we are using automated temporal planning to generate the plans as a sequence of actions with duration. To model our domain using automated temporal planning, we used **PDDL2.1** and **PDDL2.2** as modelling languages.

We develop our model with a simple railway system, to test the effectiveness of automated temporal planning. automated planners give a good result on optimization of train drive time and route choice conditions. Then we work on the Bangladesh Railway system. we almost cover all main routes of the Bangladesh Railway map. PDDL domain and problem files of our model successfully generate the optimized plan. Evaluating the experiment result, we can assure that, our model gives optimal train schedules.

Contents

List of Figures

List of Notations

ADB	Asian Development Bank
BR	Bangladesh Railway
PDDL	Planning Domain Definition Language
BRA	Bangladesh Railway Authority
DRM	Divisional Railway Manager

List of Tables

Chapter 1

Introduction

The railway system is a very significant part of the transportation system in Bangladesh. It provides a cheaper and safer travelling facility. But sometimes rail systems collapse due to accidents, routing problems, crossing etc. So, this work aims to generate schedules for the railway system to minimize the driving time and avoid collisions between trains. And another aim is to generate an optimal train schedule.

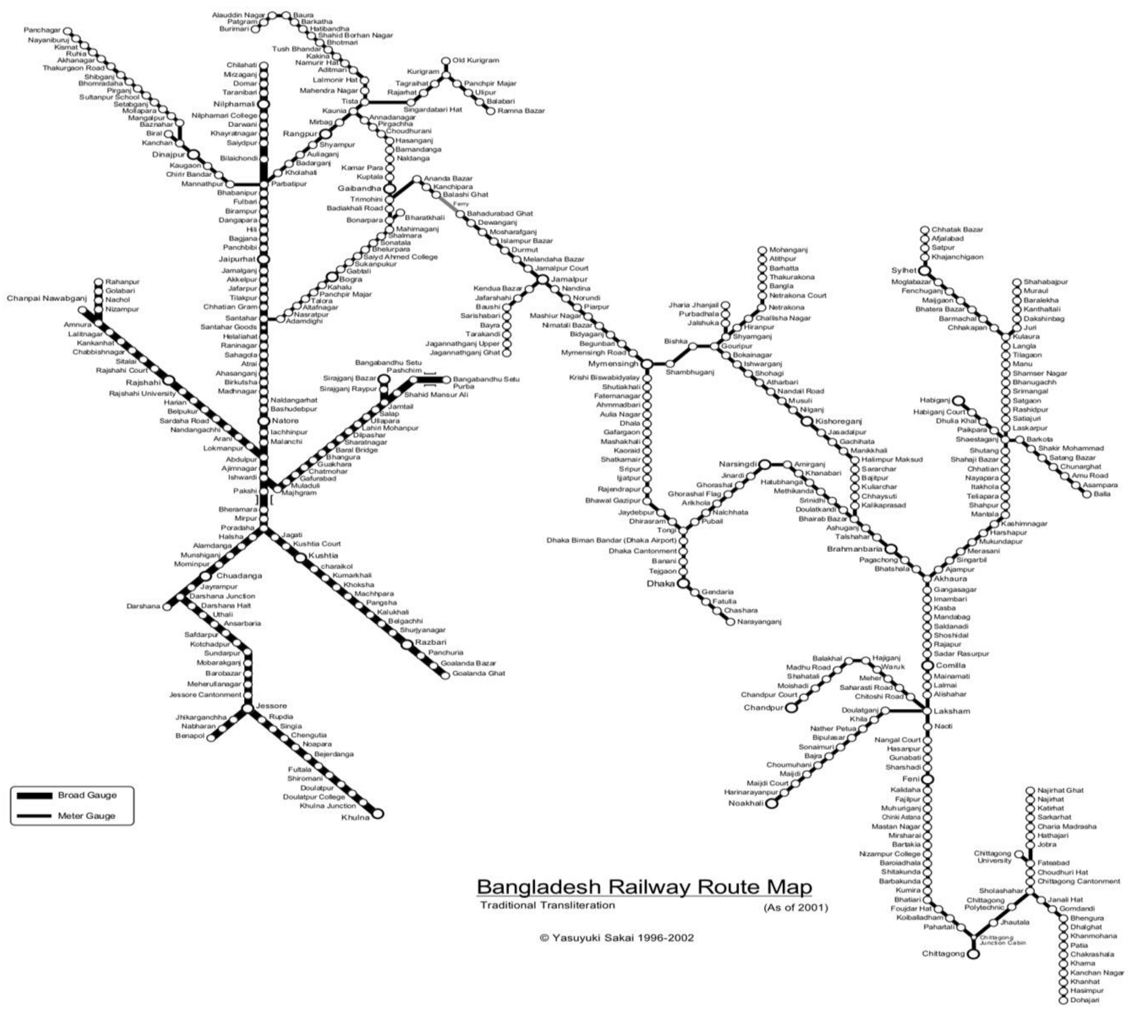


Fig-1: Bangladesh Railway Route Map

Credit: Yasuyuki Sakai 1996-2002

1.1 Motivation

Bangladesh is still a developing country. Most peoples choose cost-effective and safer transportation. Besides these other transportation like the bus is more costly and insecure than the train. People are likely to avoid bus or other road transportation services because of traffic jams and poor transportation systems due to bad road conditions.

And railway system could play a vital role to meet passenger's demands. But delays, accidents and scheduled disasters can cause a significant loss. So it is very important to make the train schedule optimal. Since train scheduling is extremely sensitive and related to a massive number of people, scheduling train becomes an attractive research area for us. Those are the main motivations for us to make an optimal train schedule.

1.2 Problem Statement

The train scheduling system is a continuous system. Train routes always stay busy. So proper scheduling is very important. Time-saving and safe journeys are major challenges in this digital age. To optimize train drive time, avoid train collisions and prevent train schedule disasters by choosing correct routes, we proposed to develop a train scheduling model using automated temporal planning.

By completing this thesis, we may expect to get the solutions to the following problem statements:

1. Can the scheduling domain model be developed using temporal planning specifications?
2. Can an automated schedule be generated using the model?
3. Is the generated schedule correct?
4. Is the generated schedule capable of optimizing the train schedule?
5. Is the model can generate schedules in optimal time?

1.3 Thesis Outline

Total 5 chapters in our thesis. Chapter 1 consists introduction to the Bangladesh Railway System, the motivation of our thesis and the problem statement of our thesis topic. Chapter 2 describe the background and the related work. Chapter 3 describes the methodology of the thesis. This chapter explained how domain models and problem models of train scheduling use automated temporal planning.

Chapter 4 explained the experimental setup and analysis of the result. Conclusion and future work explained in chapter 5.

Chapter 2

Background and Related Work

The motive of this chapter is to give a brief study of the fundamentals of the BR system and automated temporal planning. This chapter describes the background and related works of BR and automated temporal planning. In section 2.1 we describe the Bangladesh Railway system and some problems of BR. Section 2.2 is about the importance of automated planning in problem-solving and the study of temporal planning. In section 2.3 we describe the language PDDL. Section 2.4 describe temporal planning. In section 2.5 we describe some planners that could be used to find temporal plans. Section 2.6 discuss temporal planning and how can we use it to schedule train. Section 2.7 discuss some work that is similar to our work.

2.1 Bangladesh Railway System

Use Bangladesh Railways is a major transport corporation that is both state-owned and government-run. Bangladesh Railway is separated into two divisions: East and West, with two general managers in charge of each. The Director-General of Bangladesh Railways is accountable to two general managers in each of the two regions. The Railways' day-to-day operations were then separated from the Ministry and turned over to the Director-General, along with the Railway Professionals, on August 12, 1995. The 9-member Bangladesh Railway Authority (BRA), chaired by the Minister of Communications, was established to make policy decisions. All administrative and policy-making tasks are completed by the Additional Director General and the Joint Director-General. The two zones' general managers are supported by several specialist departments in charge of operations, maintenance, and financial management. Each zone is subdivided further into two primary operating divisions. The Divisional Railway Manager (DRM) oversees these departments, which are assisted by departmental officers from various specialized departments such as installation, transportation, commercial, archaeological, mechanical, way and works, signaling and telecommunications, electrical, medical, and security forces. The Government of Bangladesh created a separate Railway Department under the Ministry of Communications and later the Ministry of Railways to inspect the various activities of the Bangladesh Railways relating to train operations and to maintain the safety of rail transit [2].

2.1.1 Classes of Train

Class	Description
First Class AC	The first-class AC train is more expensive than others. The air-conditioned facility is available only on popular Inter-City routes. Besides these, this type of train has sleeping accommodation and other privacy features like personal coupes.
First Class	First-class is relatively less luxurious than first-class AC. Overall not bad at all. This type of train does not have an AC facility. But have other facilities like first-class AC.
First Class Chair	On wide-gauge railroads, the first-class chair train has five seats in a row. However, meter gauge trains have four seats in a row. These sorts of trains are typically used daily.
Second Class– Shovon Chair	Passengers in the middle class are accustomed to this style of train. The facilities are almost certainly first-class, but at a lower cost.
Second Class- Shovon	Second–Class Shovon is one of the cheapest train classes. These train seats are not comfortable as other luxurious train services.
Second- Class Shulov	Bangladesh's cheapest train service. This train seat is cushioned and made of pressed wood and steel. Short-distance routes and metropolitan locations are covered by this rail service. A seat is not guaranteed, even if a ticket is. This type of train is commonly referred to as a "local train."

2.1.2 Some Accidents that occurs in railway

Date	Description
11 July 2006	At least 32 people were killed and 40 injured when a speeding train crashed into a passenger bus at an unprotected rail crossing in Joypurhat [3].
05 February 2014	The locomotive of the Dhaka-bound Subarna Express derailed near Pahartali Loco Shed in Chittagong at 6.45 am due to a signal error. No one was injured or the train damaged. The train was rescued two hours later [4].
13 April 2014	At 3:20 am, a Dhaka-bound Ekota Express collided head-on with a Lalmonirhat-bound Lalmoni Express at Ullapara railway station in Sirajganj district. Three Ekta Express bogies and one Lalmoni Express bogy were destroyed.A total of seven coaches have left the line. At least two were killed and 50 were injured [5][6].
15 July 2019	At least 10 people were killed when a microbus collided with a Dhaka-bound train at an infamous level crossing in Ullapara upazila of Sirajganj district [7].
12 November 2019	The Dhaka-bound Turna Nishita Express hit the Chittagong-bound Udayan Express at Mandbhag railway station in Kasba upazila of Brahmanbaria district around 3am. At least 16 people were killed and 73 injured. Two bogies of the Udayan Express have been damaged [8][9].
07 November 2020	An accident occurred in Kaliakoir, Gazipur: a woman was killed and five others injured [10].
21 May 2022	An accident occurred at a level crossing in the Pubali area of Gazipur district: 3 people died in this accident [11].

2.1.3 Schedule Disasters

Scheduled disasters are very common in Bangladesh's railways. Every Eid or any occasion, the train speed decreases due to the overload of passengers. When the crowd of homebound passengers started at Kamalapur railway station, it was a disaster to maintain the schedule of the Eid special train. According to railway station sources, Eid special and local trains are usually delayed due to private ownership. During Eid, the hazards get worse as the passengers have to cope with the pressure. Passengers suffer due to long waits at the overcrowded station in the scorching heat. Many families at the station are seen sweating and waiting for the train. Moreover, all the passengers have to face the last fight while boarding the train through the crowd.

News of schedule disaster

- Eid holidaymakers face tailbacks, schedule disruptions. [12] [13]

2.1.4 Interruption for Line maintenance

Sometimes it is important to maintain the line to make it suitable for train movement. When the line is under maintenance, the train schedule is interrupted, and for this reason, the passengers suffer a lot. Last week, one of our teammates was a witness to this. He is going from Joydebpur to Dhaka and facing that problem. To solve this problem, the Silkcity Express from Rajshahi reaches Kamlapur station 3 hours late.

2.2 AI Planning

Don't Planning is selecting the steps that are required to achieve a desired goal. Planning is the basic property of intelligent behavior. It involves the use of logical thinking and imagination to visualize not only a desired goal but the steps necessary to achieve that result.

AI planning is the study of computational models and methods of creating, analyzing, managing, and executing plans [14]. Another definition can be said to be that it is a field of Artificial Intelligence which explores the process of using autonomous techniques to solve planning and scheduling problems. A planning problem is one in which we have some initial starting state, which we wish to transform into a desired goal state through the application of a set of actions.

Plan creation of AI planning is related to the state transformation problem. A plan is the ordered sequence of actions that can be taken from the initial state to the goal state. The actions are executed at particular times. State transformation problems are those characterized by an initial state, a desired goal, and a set of possible actions that can be taken to change the state. The ability of a plan depends on two capabilities:

1. The ability to predict the consequences and
2. Find the most efficient result.

There are **four types** of planning [14] These are:

1. **Classical Planning:** Classical planning is the simplest planning, which assumes a deterministic, discrete, and essentially non-temporal world model.
2. **Numeric Planning:** Numeric planning relaxes the assumption of a discrete and finite model.
3. **Temporal Planning:** Temporal planning extends the classical planning problem with scheduling plan actions in time.

4. **Hybrid Planning:** Hybrid system planning includes the problems above and in addition handles models with discrete modes and continuous change over time, by introducing continuous processes and the exogenous events that can be triggered by plan actions or by changes in the environment.

2.3 Planning Domain Definition Language (PDDL)

A formal representation language in which we model the problem is called the problem description language. An AI planning system or Planner is that which takes problem formalization or model as input then process it with some problem-solving approach and tries to reach the goal state by solving the problem. A problem can be solved by many planners, and a planner can be applied to any problem that is expressed with any modelling language. The **Planning Domain Definition Language (PDDL)** is such a problem description language. It is used in the **International Planning Competition (IPC).** PDDL is one of the most widely supported languages by the planning system [14].

Planning the action of a robot is an interesting issue that inspired AI-based planning research from the beginning. The STRIPS planning system directs the precondition effect model of classical planning.

The PDDL versions that are published are as follows [14]:

_ PDDL 1 [McDermott et al., 1998].

_ PDDL 1.2 [Bacchus, 2000]. It's excluded some unused features from the language.

_ PDDL 2.1 [Fox and Long, 2003]. It's elaborate the language to represent numeric planning and temporal planning problems. It also introduced a syntax for specifying an objective function for optimization.

_ PDDL+ [Fox and Long, 2006]. It's further extended the numeric and temporal representation to hybrid planning.

_ PDDL 2.2 [Edelkamp and Hoffmann, 2004]. It's added two features: Axioms, which add more expressive conditions to classical planning, and timed initial literals, which provide a syntactic convenience for defining a schedule of predictable events in temporal planning.

_ PDDL 3.0 [Gerevini et al., 2009]. It's added syntax for temporally extended goals and preferences to classical planning.

_ PDDL 3.1 [Helmert et al., 2008]. It's defined the limited syntax for specifying action costs.

The main purpose of modelling planning problems in PDDL is to find solution plans. http://planning.domains is such an online editor with PDDL specific features. It does not

support all PDDL, because it has a runtime limit of 15 seconds. Because of that it is restricting to solve smaller problem [14]
PDDL divides the planning problem into two parts. They are the

a. domain file and the

b. problem file.

The domain file contains the state variable (facts that may be true or false) and action. The problem file contains the initial and goal state with the combination of state variables and action of the planning problem. In PDDL, any valid name can start with a letter and contain letters, digits, hyphens (-), and underscore (_). Whitespaces are normally ignored in PDDL except that they are required to separate names, variables, keywords and other lexical elements. PDDL is case insensitive.

The ": requirements" section specifies which PDDL feature is used by the domain and what type of planning problem it is. Parameter symbols must start with a '?' There must be different parameters used in the declaration of a predicate. PDDL comments start with a semicolon (;) and end in the same line. Sometimes, for every valid move, the reverse move is also true. In our problem, first, we use this, but later on, we drop that statement with the addition of a line. We can assign each action a cost so that we can calculate the total cost. A table of all pddl2.1 requirements may be found here. Some requirements imply others, while others are shorthand for common requirement sets. If a domain does not specify any needs, it is presumed that a requirement for strips will be declared.

Requirement	**Description**
:strips	Additions and deletions in the STRIPS style
:typing	Allow type names in variable declarations.
:negative-preconditions	In goal descriptions, do not allow it. use the negation of the predicate.
:disjunctive-preconditions	In goal descriptions, allow or.
:equality	As a built-in predicate, support =.
:existential-preconditions	In goal descriptions, allow exists.
:universal-preconditions	In goal descriptions, allow forall.
:quantified-preconditions	= :existential-preconditions + :universal-preconditions
:conditional-effects	Allow when the effects are active.
:fluents	Allow assignment operators and arithmetic preconditions to be used to define functions and use effects.
:adl	= :strips + :typing + :negative-preconditions + :disjunctive-preconditions + :equality + :quantified-preconditions + :conditional-effects
:durative-actions	Allows for long-term actions.

	It's important to note that this does not imply: fluents.
:duration-inequalities	Allows inequalities to be used to control the duration of durative actions.
:continuous-effects	Allows durative actions to effect fluents in a continuous manner throughout their duration.
:predicates	hold the state variables that are binary variables that they represent facts that are either true of false.
:init	defines the initial state of the problem. It list all the variable that are true in that state. There is no and or additional parentheses used in it.
:goal	contains the condition that must be meet or fulfil at the end of a valid plan. The goal condition has the same form as an action precondition.
:objects	can list all the object that used in the problem.
:types	used to declare the names of the types. We need to specify the type to every parameter in the predicate and action declaration.

Conditional effects is used to effect on the world only under certain condition.
The structure of conditional effect is :

(*when* <*head*> <*body*>). If <head> is holds in the current state then <body> should hold in the state reached by applying this action.

As our problem is with durative actions, we need both discrete and continuous actions for our plans. Both forms of durative action structure consist of logical changes caused by the application of the action. We always think of logical change as immediate, so the continuous aspects of a continuous action only indicate how numerically values change at intervals of action. Modeling of temporary relationships in a discretized durative action is done through temporally annotated conditions and effects. All of the conditions and effects of durative actions must be temporally annotated. Whether a conditional note makes it clear the corresponding proposal must be maintained at the beginning of the interval (the point at which the action is applied), at the end of the interval (the point at which the final effects of the action are asserted), or from the beginning to the end (invariant over the duration of the action) The annotation of an effect makes it clear that the effect is immediate (it occurs at the beginning of the interval) or delayed (it occurs at the end of the interval). So any other discrete activity is performed at the identified beginning and end points of the actions in the plan. We have considered adopting the Convention, which should apply to all restrictions.

Open gaps within the start and end points as well as sustainable action, however, have been decided. The reason for this is that it would be impossible to express such a situation in reality. This open gap needs to be maintained. In our problem, the train must stay at a station for a given interval. If there is any problem, the train can skip an unimportant station. If a condition is required as a final prerequisite as well as an unchangeable condition, it means that any action which affects the unchangeable must begin after the completion of the action which requires that unchangeable condition. For example, in our problem, the train cannot move away until the time interval is finished. The objective of discrete-durative actions is to work with continuous change and focus on the end points over which change takes place. However, durable action language and semantics need to be strengthened when a plan needs to handle constantly changing values as well as discretely changing values [15].

2.4 Automated Temporal Planning

Automated planning and scheduling, sometimes known as AI planning, is a discipline of artificial intelligence that deals with the realization of plans or action sequences, usually for intelligent agents, autonomous robots, and unmanned vehicles to carry out.

The planning problem in artificial intelligence is concerned with intelligent entities such as robots, people, or computer programs making decisions in order to attain a goal. Automated temporal planning is a branch of automated planning that allows us to integrate scheduling and planning. It selects a set of actions (a plan) that begins with the initial state and progresses to the objective state in the shortest amount of time. A temporal plan can be defined as a scheduled schedule in which the sequence of actions has a beginning and end time. A domain description plus a problem description make up a typical temporal planning entity. The domain description contains the description of a certain environment (in this case, train scheduling) as well as the actions that are permissible in that environment. The problem description offers details on a specific issue that exists in that environment.

Automated temporal planning is a good option for train scheduling operations. Because of its nature, automated temporal planning may handle a variety of concerns, such as the action's preconditions and aftereffects throughout time, the plan's time span, the action length, concurrency between actions, deadlines, numeric properties, and so on. The domain environment is described using predicates and numeric functions. The predicates are a set of facts with boolean values (either true or false). The numeric functions are variables with real-number values. Numeric fluent or fluent is another term for numeric function. These predicates and numeric functions are used to model the activities (durable or immediate).

There are three aspects to a durable action: duration, conditions, and effects. The duration is self-explanatory, as it specifies how long the activity will last. The logical statement produced with predicates and numeric fluent is called the condition. For the activity to be carried out, it must be true. In terms of duration, the conditioning can be designed in three different ways. First and foremost, before the action begins (at start). Second, before the action concludes (at the conclusion), and third, for the duration of the action (overall). The effects occur after the action has been completed. In terms of time, the repercussions of the durative activity can manifest themselves in two ways. First, at the start of the action (at the start), and then at the end of the action execution (at the end).

2.5 Planner

The AI Planner package can create ideal plans for a variety of applications, including agent AI, storyline generation and management, game/simulation validation, tutorial design, automated testing, and more. The package includes a generic planner framework, writing tools, and a plan visualizer [16].

Because AI planners have evolved with the languages they utilize, different planners support different syntaxes to varying degrees. More sophisticated syntaxes, such as PDDL3.0 or PDDL+, are not supported by older planners. Older styles of syntax have been deprecated in some circumstances, and as a result, some newer planners may not support certain syntax variants. This means that a newer planner may be unable to run an older problem, and an older planner may be unable to run a newer problem in some instances. To address this issue, we'll try to provide notes on what features a wide range of planners support as this guide develops. However, there are many different types of planners, and some are objectively better or worse than others, either in general or in specific situations. To complement the focus on general-purpose syntax, this article will once again focus on the best general-purpose planners [16]**.**

Almost all AI planners are designed to work with Linux-based compilation. Certain planners have been successfully ported to Windows and Mac, and the way their source code is written in some cases makes porting them simple (such as the Java-based planner JavaFF). Planners are rarely distributed in binary form, necessitating compilation. The methods for compiling planners are not covered in this article. Any instructions on how to compile a planner are supplied for informational purposes only, and if assistance with compilation is required, the planner's designer should be contacted directly [16].

Most planners are designed to be launched from the command line once you have a compiled binary. For inputting a domain and problem file, most planners use one of these two syntaxes:

```
./<planner> <domain> <problem>
./<planner> -o <domain> -f <problem>
```

There are many planners available for temporal planning, such as, COLIN, CP4TP, POPF, LPG, LPG-TD, TFLAP, TemPorAl, popcorn-base, Baseline Optic etc. Two of them are briefly explained in the following section.

2.5.1 The LPG-*td* version (June 2004)

For implementing our problem at first we use "**The LPG-*td* version (June 2004)**" planners. LPG-td is a new version of LPG that enhances and extends version 1.2 (see below) and competed in the 4th International Planning Competition in 2004. LPG-td is an extension of LPG that supports the new "timed beginning literals" and "derived predicates" features of the standard planning domain description languages PDDL2.2. Timed beginning literals are a convenient way to represent facts that become true or untrue at specific known times. They relate to deterministic unconditional exogenous events, which are useful for modelling planning domains. Predicates that are derived are those that are unaffected by any of the planner's actions. A collection of implicative domain rules determines their truth values [17]

There are three versions of LPG-td submitted to IPC4, but due to time restrictions in producing and assessing all data, only the first two were evaluated [17]:

LPG-td .speed locates a plan as rapidly as possible and then pauses;

LPG-td .quality discovers a plan and then spends a given amount of CPU time (automatically decided) trying to enhance it; (30 min for IPC4);

LPG-td .bestquality incrementally seeks the best plan that the planner can derive within a user-specified CPU-time limit.

2.5.2 The CP4TP (Classical Planning for Temporal Planning) version

The CP4TP (Classical Planning for Temporal Planning) planner is a portfolio planner that combines the sequential planner (SEQ), TPSHE, TP, and STP temporal planners implemented in this repository. It came in second place in the International Planning Competition (IPC temporal)'s track in 2018. More information is available on the competition's official page. The portfolio's code is available here, but you can execute it by following the instructions in this repository.
Using the following command, the planners can be run in a portfolio order, from most restrictive to least restrictive (SEQ, TPSHE, TP, STP):

plan_portfolio.py [-h] [--plan-file PLANFILE] [--time TIME] [--memory MEMORY] [--generator GENERATOR] [--no-iterated] [--validate] [--no-validate] [--use-full-time] domain problem.

where:
—domain: The input domain's path
—problem: The path to the input problem.
—plan-file: The output temporal plan's name.
—time: The maximum amount of time you have to locate a solution. 1800 seconds is the default.
—memory: The maximum amount of memory that Fast Downward will require to solve a problem. 6000 MiB is the default value.
—generator: The path to the executable generator that will be used to modify the input domain and problem. Only the Allen Algebra domain requires it.
—no-iterated: Whether or not to halt after the first solution is found. It only works with TPSHE and sequential plans.
—validate, —no-validate: Whether or not the produced plan should be validated using VAL.
—should-each-planner-use-all-remaining-time: Whether or not each planner should utilize all of the remaining time.

-h: Displays help on how to use the program.

To execute the CP4TP planner, which was entered in the 2018 International Planning Competition (IPC), simply run the command above with the following arguments:
cd temporal-planning
python bin/plan_portfolio.py --no-iterated --time 1680 --memory 6000 --use-full-time

2.6 Temporal Planning and Train Scheduling

Action length and interdependencies between multiple events or actions are at the heart of the scheduling issues. The major duties in solving scheduling problems are identifying the appropriate actions, assigning a start time and length, and ensuring that resource and ordering limitations are met. Temporal planning is an extension of classical planning, in which time is scheduled and actions are planned. It can model problems using actions that have a start time, an end time, and a duration. Temporal planning can also deal with time-initial-literals (TILs), concurrency between activities, action duration restrictions, intervals between actions, compelling actions not to overlap, time windows, and so on. Temporal planning can also be used to model timeline-based challenges, which are an important part of the scheduling process. Resources can be optimized using temporal planning and a well-defined resource metric. All of these characteristics make the temporal planning technique ideal for resolving scheduling issues.

In train scheduling, we need to schedule trains with assigning initial station, valid moving path between stations, line-status: to check whether the line is free or busy, line-maintenance mode: if the line is displaced or any accident occurs, then the specific line will be in maintenance mode and each train will have to pass through this line should wait until the maintenance period is over, speed of each train, stoppage stations of each train during its traveling time, and the distance between adjacent stations. Temporal planning is to schedule the time and plan the action, so it is effective for us to use temporal planning.

2.7 Related Work

Earlier of this paper, we describe the fitness of temporal planning for scheduling the operation of train scheduling. This section introduce us with some work which has similarities to our work.

2.7.1 Temporal Planning with Continuous Change

Because of our interest in train scheduling with temporal planning, "Temporal Planning with Continuous Change" by J. Scott Penberthy and Daniel S. Weld is closest to our work. This work is related to the plane schedule. If a plane moves passengers between cities, the paper offers ZENO, a least commitment planner for actions that occur over long periods of time. Goals with a deadline, metric preconditions, metric impacts, and constant change are all encouraged. Simultaneous activities are permitted as long as their results do not conflict.

Unlike other complex language planners, the ZENO planning algorithm is sound and comprehensive.

The running code is a full implementation of the formal method that can solve basic problems (those with fewer than a dozen stages). If the plane is flying slowly, which means it travels at 400 miles per hour and takes 1 gallon of fuel every 2 miles, on average. If the plane is flying fast, which means it travels at 600 miles per hour and takes 1 gallon of fuel every 3 miles, on average. Passengers can be boarded in 30 minutes and deplaned in 20 minutes. Refueling gradually increases the fuel level to a maximum of 750 gallons, taking one hour from an empty tank. Boarding, deplaning, and refueling must all occur while the plane is on the ground. The plane flies routes between four cities [18].

2.7.2 In-Station Train Dispatching: A PDDL+ Planning Approach.

Another work that is closely related to our work is "In-Station Train Dispatching: A PDDL+ Planning Approach" by Matteo Cardellini,1 Marco Maratea,1 Mauro Vallati,2 Gianluca Boleto,1 Luca Oneto, DIBRIS, University of Genoa, Italy. In general, the trains halt in a station. It is the most critical work to manage the train to come to a station, stop according to the official timetable, adjust delays for other trains, and dispatch from the station to its routes. Any delay to the train must be adjusted in such a way that it can have a good effect on the rest of the network. In this paper they describe an approach to performing in-station dispatching of trains that is optimal to all trains and the rest of the network [15].

2.7.3 A Study of the Dhaka- Narayanganj Railway Route.

Another close work is "A Study of the Dhaka- Narayanganj Railway Route" by MD. Moinul Islam. Narayanganj is Bangladesh's industrial and commercial centre. It is a fast expanding metropolis. Because it is one of the most industrialized zones and has greater contact with Dhaka, people's concentration is expanding. Bangladesh's transportation system consists of roads, railways, waterways, and aviation, with railways and buses being the two most popular land transport modes. The route between Narayanganj and Dhaka is vital for trade, commerce, and industry. Bus service is more prevalent along this route. However, Dhaka's bus service has several flaws. Too many buses on the road cause traffic congestion, pollution, and accidents, among other things. In the past, the railway was a very popular means of transportation for many reasons. Railways have grown in popularity as a means of public transportation around the world. However, the contrary appears to be true in Bangladesh, particularly in Dhaka. And this necessitates an inquiry into why this is the case. Thus, the specific goals of this research were to:

(i) investigate the problems that have rendered the railway inefficient and unreliable over time;

(ii) determine how this can be reversed; and

(iii) determine why passengers travelling from Dhaka to Narayanganj or vice versa ride buses and whether they will travel by rail if reliability and efficiency are assured.

According to this study, train service is far superior to bus service and is also far more reliable. The study also uncovered the comparative advantages and drawbacks of train service. In addition, a list of recommendations for popularizing the railway service has been proposed. In this regard, the policy proposals presented in this thesis can be considered when drafting a future growth strategy [20].

2.7.4 Algorithmic decision support for train scheduling in a large and highly utilized railway network.

Another work is "Algorithmic decision support for train scheduling in a large and highly utilized railway network" by Caimi, Gabrio C. In his work, he looks at the topic of establishing train schedules in general, with a special emphasis on large, highly used train networks. To this end, a methodology has been devised that allows train schedules to be automatically generated from a given train service intention, which is the description of the train services that passenger and freight rail companies would like to offer to the customers. The aim is to create specific, conflict-free rail paths for each train that meet the commercial requirements for the timetable, which are presumed to be specified. The thesis develops a comprehensive approach that starts with a commercial description of proposed railway services and ends with a conflict-free detailed itinerary for an entire day. The system employs a divide-and-conquer strategy based on three levels of description: service intent, macro-timetable, and microscopic schedule. The tiers are connected in such a way that planners can intervene in specifications at any level, as well as create feedback loops for experimenting with different scenarios [21].

2.8 Summary

This section describes the background and related work of train scheduling with automated temporal planning. We will go over temporal planning in detail in the following chapter.

Chapter 3

Methodology

This The work methodology is explained briefly in this chapter. The environment set up, how we install the planners and design of the domain, and the problem are described briefly in this section. First, "Train Scheduling and Planning Model Development" discussed the railway system and some complex decision-making problems in train scheduling. "Domain Design" explains how we design domains. "Problem design" explains how we generate problem files.

"Plan Metrics" explains the metric function. "Planner" explained the planning and installation process of it. Then finally, "Plan Validation," or how we validate a plan.

3.1 Train Scheduling and Planning Model Development

A railway system is a very complex system. It has many complex decision-making problems like route choice, crossing, waiting and stopping, etc. Line maintenance and accidents are also an obstacle to the scheduled train system. It is a very challenging task to schedule the operation of a train system. This section explains how we design the scheduling model using automated temporal planning.

3.2 Domain Design

When it comes to planning and scheduling issues, domain design is crucial. The train, line, and station are the most significant aspects of our scheduling problem. If the line is not free, a train cannot move from one station to another. If the line is free, the move is valid. Train-at, visited, valid-move, stoppage-at, and free-line are the five predicates we created. Also, take three durative-actions to address real-world issues. Durative-actions are drive, wait, and stop.

3.2.1 Temporal Action drive-train

```
( (:durative-action drive-train
        :parameters (?t - train ?from ?to - station ?l - line)
        :duration (= ?duration (/(station-distance ?from ?to ) (train-speed ?t)))
```

```
    :condition (and (at start (train-at ?t ?from))
            (at start (valid-move ?from ?to ?l))
            (over all (free-line ?l))
            (over all (not (maintenance-line ?l)))

    )
    :effect(and(at start (not (train-at ?t ?from)))
            (at start (not (free-line ?l)))
            (at end (train-at ?t ?to))
            (at end (free-line ?l))
            (at end (increase (total-cost) (* ?duration 1)))

    )
)
```

In this durative-action train, station and line are the parameters. A train can make a valid move only if the line free condition is true. Duration of the action is calculated by the driving time with respect to the distance between two stations and the speed of the train. When the drive action is executed, there will be the following effects:

-Train will move to the next station and not be at the previous station.
-At the start line, it will be busy to avoid collision with other trains.
-The train reaches the next station and
-Line will be free again.
-Cost will increase with duration.

3.2.2 Temporal Action stop-train:

```
(:durative-action stop-train
    :parameters (?t - train ?s - station)
    :duration (= ?duration 2)
    :condition (and
            (over all (train-at ?t ?s))
            (at start (stoppage-at ?t ?s))
    )
    :effect (and
            (at start (not (stoppage-at ?t ?s)))
            (at end (visited ?t ?s))
            (at end (increase (total-cost) (* ?duration 0)))
    )
)
```

Train and station are parameters in this durative-action. Duration time is 2 (fixed by duration). The action is executed after checking the train location. If the train is at its stopping station, then the train will stop.
After execution, there will be the following effect on our proposed plan:

-Station is visited and the train stopped.
-The train is not at a stoppage.
-Increase the total cost with duration.

3.2.3 Temporal Action maintenance-for

During the maintenance period of a line, no train can pass through this line and trains need to wait at the nearest station until the maintenance period is over.

```
(:durative-action maintenance-for
        :parameters (?l - line ?t - train ?from - station ?to - station)
        :duration (= ?duration 3)
        :condition (and
                (at start (train-at ?t ?from))
                (at start (maintenance-line ?l))
                (at start (valid-move ?from ?to ?l))

        )
        :effect (and
                (at end (not(maintenance-line ?l))
        ))
        (at end (increase (total-cost) (* ?duration 0)))
        )
)
```

In this durative-action station, train and line are the parameters. The default waiting time is 3.

After checking the train location, maintenance and line condition, the action will be executed. After execution, the following effects will be there:

-Maintenance mode is over and the line will be free.

3.3 Problem design

Here we designed a problem model of a train scheduling problem. We prepared the initial state, valid moves, distance between two adjacent stations, and goal. In the initial state, we assume that some trains are at stations. The goal is to reach specific other stations. Valid moves, free-lines, and distance are given. The lines that will be in maintenance mode are also given. We also created a problem generator file called "Plan.sh" that can design problem files automatically based on input. In our testing problem, we take 15 different trains, for a total of 57 rail lines and 47 stations. A simplified map of BR with routes and stations is given on the next page.

Now, an automated temporal planner will generate an optimal plan schedule to reach the destination stations.

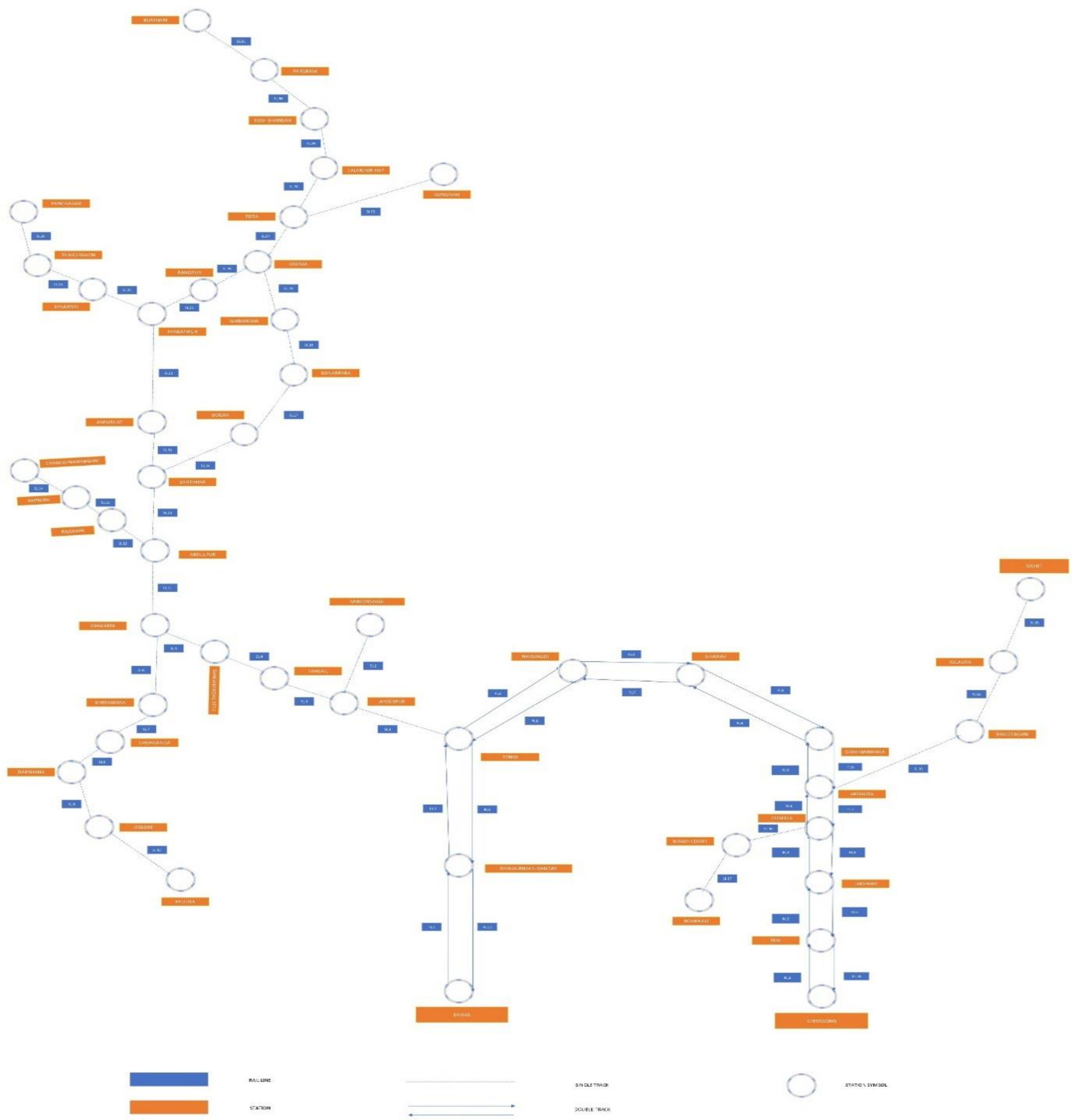


Fig-2

```
(define (problem train-scheduling)
	(:domain temporal-train-schedule)
	(:objects
		;Train Name
			suborno-express truna-express bonolota-express ekota-express
		drutojan-express chitra-express sundorban-express paharika-express
		upobon-express titumir-express rupsha-express kurigram-express
		kapataksha-express udayan-express upokul-express - train
		;Station Name
			Dhaka Dhaka-Biman-Bandar Tongi Narsingdi Bhairab
		Brahmanbaria Akhaura Cumilla Laksham Feni Chittagong - station
		Maijdi-Court Noakhali Shaestaganj Kulaura Sylhet Joydebpur
		Mymensingh Tangail Bangabandhu-Setu Ishwardi Khulna Abdulpur
		Rajshahi - station
			Bheramara Chuadanga Darshana Jessore Amnura
		Chanpainawabganj Santahar Jaypurhat Parbatipur Bogra Bonapara
		Gaibanda - station
			Dinajpur Thakurgaon Panchagar Rangpur Kaunia Tista Kurigram
		Lalmonir-Hat Tush-Bhandar Patgram Burimari - station
			;Forward Line
			fl1 fl2 fl3 fl4 fl5 fl6 fl7 fl8 fl9 fl10 - forward-line
			;Reverse Line
			rl1 rl2 rl3 rl4 rl5 rl6 rl7 rl8 rl9 rl10 - reverse-line
			;Single Line
			sl1 sl2 sl3 sl4 sl5 sl6 sl7 sl8 sl9 sl10 sl11 sl12 sl13 sl14 sl15 sl16
			sl17 sl18 sl19 sl20 sl21 sl22 sl23 sl24 sl25 sl26 sl27 - single-line
			sl28 sl29 sl30 sl31 sl32 sl33 sl34 sl35 sl36 sl37 - single-line

	)

	(:init


		;Forward Move
		(valid-move Dhaka Dhaka-Biman-Bandar fl1)
		(valid-move Dhaka-Biman-Bandar Tongi fl2)
		(valid-move Tongi Narsingdi fl3)
		(valid-move Narsingdi Bhairab fl4)
		(valid-move Bhairab Brahmanbaria fl5)
		(valid-move Brahmanbaria Akhaura fl6)
		(valid-move Akhaura Cumilla fl7)
		(valid-move Cumilla Laksham fl8)
```

```
(valid-move Laksham Feni fl9)
(valid-move Feni Chittagong fl10)

(valid-move Tongi Joydebpur sl1)
(valid-move Joydebpur Mymensingh sl2)
(valid-move Joydebpur Tangail sl3)
(valid-move Tangail Bangabandhu-Setu sl4)
(valid-move Bangabandhu-Setu Ishwardi sl5)
(valid-move Ishwardi Bheramara sl6)
(valid-move Bheramara Chuadanga sl7)
(valid-move Chuadanga Darshana sl8)
(valid-move Darshana Jessore sl9)
(valid-move Jessore Khulna sl10)
(valid-move Ishwardi Abdulpur sl11)
(valid-move Abdulpur Rajshahi sl12)
(valid-move Rajshahi Amnura sl13)

(valid-move Amnura Chanpainawabganj sl14)
(valid-move Abdulpur Santahar sl15)
(valid-move Santahar Bogra sl16)
(valid-move Bogra Bonapara sl17)
(valid-move Bonapara Gaibanda sl18)
(valid-move Gaibanda Kaunia sl19)
(valid-move Santahar Jaypurhat sl20)
(valid-move Jaypurhat Parbatipur sl21)
(valid-move Parbatipur Dinajpur sl22)
(valid-move Dinajpur Thakurgaon sl23)
(valid-move Thakurgaon Panchagar sl24)
(valid-move Parbatipur Rangpur sl25)
(valid-move Rangpur Kaunia sl26)

(valid-move Kaunia Tista sl27)
(valid-move Tista Lalmonir-Hat sl28)
(valid-move Lalmonir-Hat Tush-Bhandar sl29)
(valid-move Tush-Bhandar Patgram sl30)
(valid-move Patgram Burimari sl31)
(valid-move Tista Kurigram sl32)
(valid-move Akhaura Shaestaganj sl33)
(valid-move Chanpainawabganj Kulaura sl34)
(valid-move Kulaura Sylhet sl35)
(valid-move Cumilla Maijdi-Court sl36)
(valid-move Maijdi-Court Noakhali sl37)
```

```
;Reverse Move
(valid-move Chittagong Feni rl1)
(valid-move Feni Laksham rl2)
(valid-move Laksham Cumilla rl3)
(valid-move Cumilla Akhaura rl4)
(valid-move Akhaura Brahmanbaria rl5)
(valid-move Brahmanbaria Bhairab rl6)
(valid-move Bhairab Narsingdi rl7)
(valid-move Narsingdi Tongi rl8)
(valid-move Tongi Dhaka-Biman-Bandar rl9)
(valid-move Dhaka-Biman-Bandar Dhaka rl10)

(valid-move Joydebpur Tongi sl1)
(valid-move Mymensingh Joydebpur sl2)
(valid-move Tangail Joydebpur sl3)
(valid-move Bangabandhu-Setu Tangail sl4)
(valid-move Ishwardi Bangabandhu-Setu sl5)
(valid-move Bheramara Ishwardi sl6)
(valid-move Chuadanga Bheramara sl7)
(valid-move Darshana Chuadanga sl8)
(valid-move Jessore Darshana sl9)
(valid-move Khulna Jessore sl10)

(valid-move Abdulpur Ishwardi sl11)
(valid-move Rajshahi Abdulpur sl12)
(valid-move Amnura Rajshahi sl13)
(valid-move Chanpainawabganj Amnura sl14)
(valid-move Santahar Abdulpur sl15)
(valid-move Bogra Santahar sl16)
(valid-move Bonapara Bogra sl17)
(valid-move Kaunia Bonapara sl18)
(valid-move Jaypurhat Santahar sl19)
(valid-move Bheramara Chuadanga sl20)
(valid-move Parbatipur Jaypurhat sl21)
(valid-move Dinajpur Parbatipur sl22)
(valid-move Thakurgaon Dinajpur sl23)
(valid-move Panchagar Thakurgaon sl24)
(valid-move Rangpur Parbatipur sl25)
(valid-move Kaunia Rangpur sl26)

(valid-move Tista Kaunia sl27)
(valid-move Lalmonir-Hat Tista sl28)
```

(valid-move Tush-Bhandar Lalmonir-Hat sl29)
(valid-move Patgram Tush-Bhandar sl30)
(valid-move Burimari Patgram sl31)
(valid-move Kurigram Tista sl32)
(valid-move Shaestaganj Akhaura sl33)
(valid-move Kulaura Shaestaganj sl34)
(valid-move Sylhet Kulaura sl35)
(valid-move Maijdi-Court Cumilla sl36)
(valid-move Noakhali Maijdi-Court sl37)

;line-status

(free-line fl1)
(free-line fl2)
(free-line fl3)
(free-line fl4)
(free-line fl5)
(free-line fl6)
(free-line fl7)
(free-line fl8)
(free-line fl9)
(free-line fl10)

(free-line rl1)
(free-line rl2)
(free-line rl3)
(free-line rl4)
(free-line rl5)
(free-line rl6)
(free-line rl7)
(free-line rl8)
(free-line rl9)
(free-line rl10)

(free-line sl1)
(free-line sl2)
(free-line sl3)
(free-line sl4)
(free-line sl5)
(free-line sl6)

```
(free-line sl7)
(free-line sl8)
(free-line sl9)
(free-line sl10)
(free-line sl11)
(free-line sl12)


(free-line sl13)
(free-line sl14)
(free-line sl15)
(free-line sl16)
(free-line sl17)
(free-line sl18)
(free-line sl19)
(free-line sl20)
(free-line sl21)
(free-line sl22)


(free-line sl23)
(free-line sl24)
(free-line sl25)
(free-line sl26)
(free-line sl27)
(free-line sl28)
(free-line sl29)
(free-line sl30)
(free-line sl31)
(free-line sl32)
(free-line sl33)
(free-line sl34)
(free-line sl35)
(free-line sl36)
(free-line sl37)

;station-distance in meters

(= (station-distance Dhaka-Biman-Bandar Dhaka)60)
(= (station-distance Tongi Dhaka-Biman-Bandar)70)
(= (station-distance Narsingdi Tongi)50)
(= (station-distance Bhairab Narsingdi)80)
(= (station-distance Brahmanbaria Bhairab)75)
```

```
(= (station-distance Akhaura Brahmanbaria)55)
(= (station-distance Cumilla Akhaura)28)
(= (station-distance Laksham Cumilla)25)
(= (station-distance Laksham Feni)90)
(= (station-distance Feni Chittagong)52)

(= (station-distance Tongi Joydebpur)20)
(= (station-distance Joydebpur Mymensingh)40)
(= (station-distance Joydebpur Tangail)40)
(= (station-distance Tangail Bangabandhu-Setu)60)
(= (station-distance Bangabandhu-Setu Ishwardi)30)
(= (station-distance Ishwardi Bheramara)95)
(= (station-distance Bheramara Chuadanga)25)
(= (station-distance Chuadanga Darshana)80)
(= (station-distance Darshana Jessore)75)

(= (station-distance Jessore Khulna)45)
(= (station-distance Ishwardi Abdulpur)20)
(= (station-distance Abdulpur Rajshahi)70)
(= (station-distance Rajshahi Amnura)50)
(= (station-distance Amnura Chanpainawabganj)65)
(= (station-distance Abdulpur Santahar)55)
(= (station-distance Santahar Bogra)40)
(= (station-distance Bogra Bonapara)30)
(= (station-distance Bonapara Gaibanda)40)
(= (station-distance Gaibanda Kaunia)20)
(= (station-distance Santahar Jaypurhat)35)
(= (station-distance Jaypurhat Parbatipur)60)
(= (station-distance Parbatipur Dinajpur)26)
(= (station-distance Dinajpur Thakurgaon)47)
(= (station-distance Thakurgaon Panchagar)50)
(= (station-distance Parbatipur Rangpur)36)
(= (station-distance Rangpur Kaunia)61)
(= (station-distance Kaunia Tista)47)
(= (station-distance Tista Lalmonir-Hat)36)
(= (station-distance Lalmonir-Hat Tush-Bhandar)45)
(= (station-distance Tush-Bhandar Patgram)55)
(= (station-distance Patgram Burimari)68)
(= (station-distance Tista Kurigram)88)
(= (station-distance Akhaura Shaestaganj)35)
(= (station-distance Shaestaganj Kulaura)40)
(= (station-distance Kulaura Sylhet)30)
(= (station-distance Cumilla Maijdi-Court)42)
```

```
(= (station-distance Maijdi-Court Noakhali)80)

; Reverse station-distance
(= (station-distance Dhaka Dhaka-Biman-Bandar)60)
(= (station-distance Dhaka-Biman-Bandar Tongi)70)
(= (station-distance Tongi Narsingdi)50)
(= (station-distance Narsingdi Bhairab)80)
(= (station-distance Bhairab Brahmanbaria)75)
(= (station-distance Brahmanbaria Akhaura)55)
(= (station-distance Akhaura Cumilla)28)
(= (station-distance Cumilla Laksham)25)
(= (station-distance Laksham Feni)90)
(= (station-distance Feni Chittagong)52)

(= (station-distance Joydebpur Tongi)20)
(= (station-distance Mymensingh Joydebpur)40)
(= (station-distance Tangail Joydebpur)40)
(= (station-distance Bangabandhu-Setu Tangail)60)
(= (station-distance Ishwardi Bangabandhu-Setu)30)
(= (station-distance Bheramara Ishwardi)95)
(= (station-distance Chuadanga Bheramara)25)
(= (station-distance Darshana Chuadanga)80)
(= (station-distance Jessore Darshana)75)

(= (station-distance Khulna Jessore)45)
(= (station-distance Abdulpur Ishwardi)20)
(= (station-distance Rajshahi Abdulpur)70)
(= (station-distance Amnura Rajshahi)50)
(= (station-distance Chanpainawabganj Amnura)65)
(= (station-distance Santahar Abdulpur)55)
(= (station-distance Bogra Santahar)40)
(= (station-distance Bonapara Bogra)30)
(= (station-distance Gaibanda Bonapara)40)
(= (station-distance Kaunia Gaibanda)20)
(= (station-distance Jaypurhat Santahar)35)
(= (station-distance Parbatipur Jaypurhat)60)
(= (station-distance Dinajpur Parbatipur)26)
(= (station-distance Thakurgaon Dinajpur)47)
(= (station-distance Panchagar Thakurgaon)50)
(= (station-distance Rangpur Parbatipur)36)
(= (station-distance Kaunia Rangpur)61)
(= (station-distance Tista Kaunia)47)
(= (station-distance Lalmonir-Hat Tista)36)
```

```
        (= (station-distance Tush-Bhandar Lalmonir-Hat)45)
        (= (station-distance Patgram Tush-Bhandar)55)
        (= (station-distance Burimari Patgram)68)
        (= (station-distance Kurigram Tista)88)
        (= (station-distance Shaestaganj Akhaura)35)
        (= (station-distance Kulaura Shaestaganj)40)
        (= (station-distance Sylhet Kulaura)30)
        (= (station-distance Maijdi-Court Cumilla)42)
        (= (station-distance Noakhali Maijdi-Court)80)



(maintenance-line sl37)

        (=(train-speed upokul-express)10)
        (train-at upokul-express Dhaka)
        (visited upokul-express Dhaka)
        (stoppage-at upokul-express Dhaka-Biman-Bandar)
        (stoppage-at upokul-express Narsingdi)
        (stoppage-at upokul-express Bhairab)
        (stoppage-at upokul-express Brahmanbaria)
        (stoppage-at upokul-express Akhaura)
        (stoppage-at upokul-express Cumilla)
        (stoppage-at upokul-express Maijdi-Court)
        (stoppage-at upokul-express Noakhali)

        (=(train-speed drutojan-express)10)
        (train-at drutojan-express Dhaka)
        (visited drutojan-express Dhaka)
        (stoppage-at drutojan-express Tangail)
        (stoppage-at drutojan-express Bangabandhu-Setu)
        (stoppage-at drutojan-express Ishwardi)
        (stoppage-at drutojan-express Joydebpur)
        (stoppage-at drutojan-express Panchagar)
        (stoppage-at drutojan-express Dhaka-Biman-Bandar)
        (stoppage-at drutojan-express Dhaka)
)
(:goal
        (and(visited upokul-express Dhaka-Biman-Bandar)
        (visited upokul-express Narsingdi)
        (visited upokul-express Bhairab)
        (visited upokul-express Brahmanbaria)
        (visited upokul-express Akhaura)
```

```
        (visited upokul-express Cumilla)
        (visited upokul-express Maijdi-Court)
        (visited upokul-express Noakhali)
        (visited drutojan-express Tangail)
        (visited drutojan-express Bangabandhu-Setu)
        (visited drutojan-express Ishwardi)
        (visited drutojan-express Joydebpur)
        (visited drutojan-express Panchagar)
        (visited drutojan-express Dhaka-Biman-Bandar)
        (visited drutojan-express Dhaka)
    ))
(:metric minimize (total-cost))
)
```

3.4 Plan Metrics

The goals of the objectives function are modeled as plan metrics. The automated temporal planner generates the plan according to plan metrics. We used the following metrics in our work:

1. (:metric minimize (total-cost))

3.5 Planner

We chose LPG-TD as the automated temporal planner to generate the plan schedule. LPG-TD can handle most of the PDDL 2.1 likes numeric fluents, durative actions, continuous effects and negative preconditions etc [22]. and all the features of PDDL2.2. LPG-TD planner can also give us an optimal plan when quality is more important than solutions. It can be operated in three modes. These are speed, quality, and n [the number of plans we want to generate]. LPG-TD keeps searching for a better solution, also known as plan schedule, as long as time and memory permits although it finds the first solution. That's why we chose the LPG-TG planner in our work to generate optimal train schedules.

3.5.1 Installation

Installation process of LPG-td [22]***:***
First, download the LPG-td planner, https://lpg.unibs.it/lpg/
Then, copy lpgtd-linux.tar.gz into the directory where you want to have the planner.

-From this directory, run the following code:

gunzip lpgtd-linux.tar.gz
tar xvf pgtd-linux.tar

To test the plan, run the following script in a terminal:

./lpg-td -o op-file-name -f p-file-name -n 3
where op-file-name represents the domain file, p-file-name represents the problem file, and -n represents the desired number of solutions.

Before running the planner, you need to know about the basic and necessary settings.
Check here for details: https://github.com/fawcettc/planning-features/blob/master/lpg/README-LPGTD

Additionally, to run LPG-td, Singularity and GO are needed.
To install Singularity and Go on your machine, visit:
https://github.com/sylabs/singularity/blob/main/INSTALL.md

3.6 Model Evaluation

We evaluate our scheduling model domain with respect to our problem statement in the previous section. Our first challenge was whether the domain model could be developed using temporal specifications. The second question was, "Can an automated schedule be generated using the model?" The third question was, "Is the generated schedule correct?" The fourth question was whether the generated schedule could optimize train scheduling, and the final question or challenge was whether the domain model could generate a schedule at the optimal time.

3.7 Generated Plan

; Version LPG-td-1.4
; Seed 68031938
; Command line: ./lpg-td -o domain_temporal.pddl -f problem_temporal.pddl -quality –timesteps -out train_plan
; Problem problem_temporal.pddl
; Time 1.01
; Plan generation time 0.55
; Search time 0.54
; Parsing time 0.01
; Mutex time 0.00
; MetricValue 118.50

0.0003: (DRIVE-TRAIN DRUTOJAN-EXPRESS DHAKA DHAKA-BIMAN-BANDAR FL1) [6.0000]
6.0005: (STOP-TRAIN DRUTOJAN-EXPRESS DHAKA-BIMAN-BANDAR) [2.0000]

8.0007: (DRIVE-TRAIN DRUTOJAN-EXPRESS DHAKA-BIMAN-BANDAR TONGI FL2) [7.0000]
15.0010: (DRIVE-TRAIN DRUTOJAN-EXPRESS TONGI JOYDEBPUR SL1) [2.0000]
17.0012: (STOP-TRAIN DRUTOJAN-EXPRESS JOYDEBPUR) [2.0000]
6.0015: (DRIVE-TRAIN UPOKUL-EXPRESS DHAKA DHAKA-BIMAN-BANDAR FL1) [6.0000]
12.0017: (STOP-TRAIN UPOKUL-EXPRESS DHAKA-BIMAN-BANDAR) [2.0000]
15.0020: (DRIVE-TRAIN UPOKUL-EXPRESS DHAKA-BIMAN-BANDAR TONGI FL2) [7.0000]
22.0023: (DRIVE-TRAIN UPOKUL-EXPRESS TONGI NARSINGDI FL3) [5.0000]
27.0025: (STOP-TRAIN UPOKUL-EXPRESS NARSINGDI) [2.0000]
29.0028: (DRIVE-TRAIN UPOKUL-EXPRESS NARSINGDI BHAIRAB FL4) [8.0000]
37.0030: (STOP-TRAIN UPOKUL-EXPRESS BHAIRAB) [2.0000]
39.0033: (DRIVE-TRAIN UPOKUL-EXPRESS BHAIRAB BRAHMANBARIA FL5) [7.5000]
46.5035: (STOP-TRAIN UPOKUL-EXPRESS BRAHMANBARIA) [2.0000]
48.5037: (DRIVE-TRAIN UPOKUL-EXPRESS BRAHMANBARIA AKHAURA FL6) [5.5000]
54.0040: (STOP-TRAIN UPOKUL-EXPRESS AKHAURA) [2.0000]
56.0042: (DRIVE-TRAIN UPOKUL-EXPRESS AKHAURA CUMILLA FL7) [2.8000]
58.8045: (STOP-TRAIN UPOKUL-EXPRESS CUMILLA) [2.0000]
60.8047: (DRIVE-TRAIN UPOKUL-EXPRESS CUMILLA MAIJDI-COURT SL36) [4.2000]
65.0050: (MAINTENANCE-FOR SL37 UPOKUL-EXPRESS MAIJDI-COURT NOAKHALI) [3.0000]
65.0052: (STOP-TRAIN UPOKUL-EXPRESS MAIJDI-COURT) [2.0000]
68.0055: (DRIVE-TRAIN UPOKUL-EXPRESS MAIJDI-COURT NOAKHALI SL37) [8.0000]
76.0058: (STOP-TRAIN UPOKUL-EXPRESS NOAKHALI) [2.0000]
19.0060: (DRIVE-TRAIN DRUTOJAN-EXPRESS JOYDEBPUR TANGAIL SL3) [4.0000]
23.0063: (STOP-TRAIN DRUTOJAN-EXPRESS TANGAIL) [2.0000]
25.0065: (DRIVE-TRAIN DRUTOJAN-EXPRESS TANGAIL BANGABANDHU-SETU SL4) [6.0000]
31.0068: (STOP-TRAIN DRUTOJAN-EXPRESS BANGABANDHU-SETU) [2.0000]
33.0070: (DRIVE-TRAIN DRUTOJAN-EXPRESS BANGABANDHU-SETU ISHWARDI SL5) [3.0000]
36.0073: (STOP-TRAIN DRUTOJAN-EXPRESS ISHWARDI) [2.0000]
38.0075: (DRIVE-TRAIN DRUTOJAN-EXPRESS ISHWARDI ABDULPUR SL11) [2.0000]

Generated Plan

40.0078: (DRIVE-TRAIN DRUTOJAN-EXPRESS ABDULPUR SANTAHAR SL15) [5.5000]
45.5080: (DRIVE-TRAIN DRUTOJAN-EXPRESS SANTAHAR JAYPURHAT SL20) [3.5000]
49.0083: (DRIVE-TRAIN DRUTOJAN-EXPRESS JAYPURHAT PARBATIPUR SL21) [6.0000]
55.0085: (DRIVE-TRAIN DRUTOJAN-EXPRESS PARBATIPUR RANGPUR SL25) [3.6000]
58.6087: (DRIVE-TRAIN DRUTOJAN-EXPRESS RANGPUR PARBATIPUR SL25) [3.6000]
62.2090: (DRIVE-TRAIN DRUTOJAN-EXPRESS PARBATIPUR DINAJPUR SL22) [2.6000]
64.8092: (DRIVE-TRAIN DRUTOJAN-EXPRESS DINAJPUR THAKURGAON SL23) [4.7000]
69.5095: (DRIVE-TRAIN DRUTOJAN-EXPRESS THAKURGAON PANCHAGAR SL24) [5.0000]
74.5098: (STOP-TRAIN DRUTOJAN-EXPRESS PANCHAGAR) [2.0000]

We validated the first three questions using a validation tool called VAL [23]. VAL tools verify domain development is correct according to PDDL 2.2 and the PDDL 2.2 specification. The plan schedule we generated is correct. Forth answer, using the automated temporal planner, our domain mode can generally optimally solve problems using quality operation mode. Finally, the domain model can generate a schedule in optima time using speed operation mode. But the generated plan is not the optimal solution.

3.8 Summary

This chapter explains the working process. It explains briefly how we design domain and problem files step by step. The domain consists of 3 functions: drive-train, stop-train, and maintenance-for. In our testing problem, we take 15 different trains, for a total of 57 rail lines and 47 stations. This chapter also describes the plan metrics, planner, and the installation process of the planner. And at last, the model evaluation is described.

Chapter 4

Result and Discussions

This chapter presents the results of an analysis of train scheduling using temporal planning. The train scheduling domain was developed using PDDL 2.1 and PDDL 2.2. We used the LPG-TD automated temporal planner for simulation purposes. Because, as we previously stated, LPG-TD can handle both PDDL2.1 and PDDL2.2.

4.1 Planning Timeline

We have designed a simple planning timeline based on the output plan.

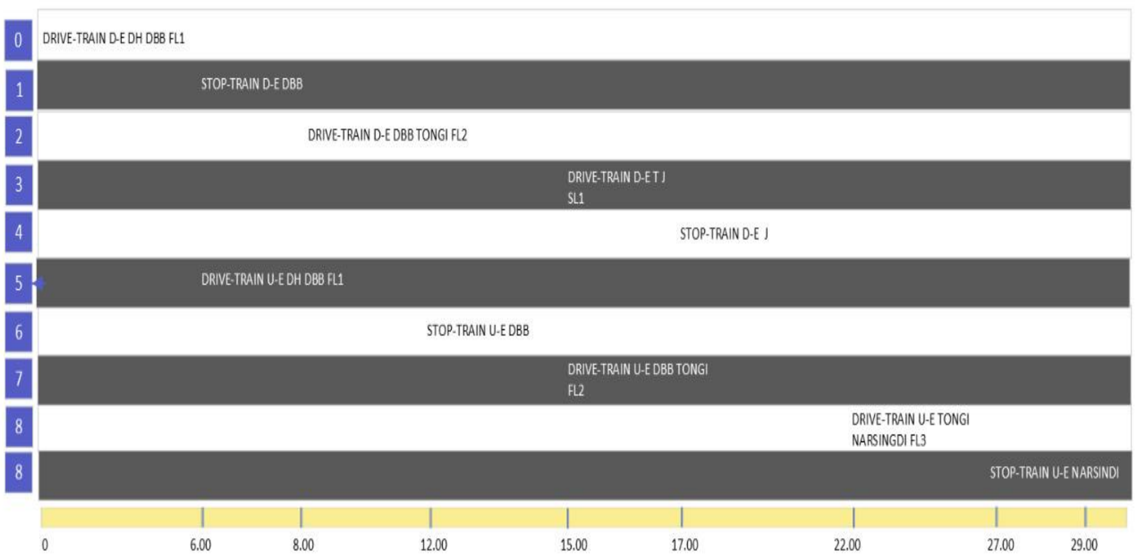


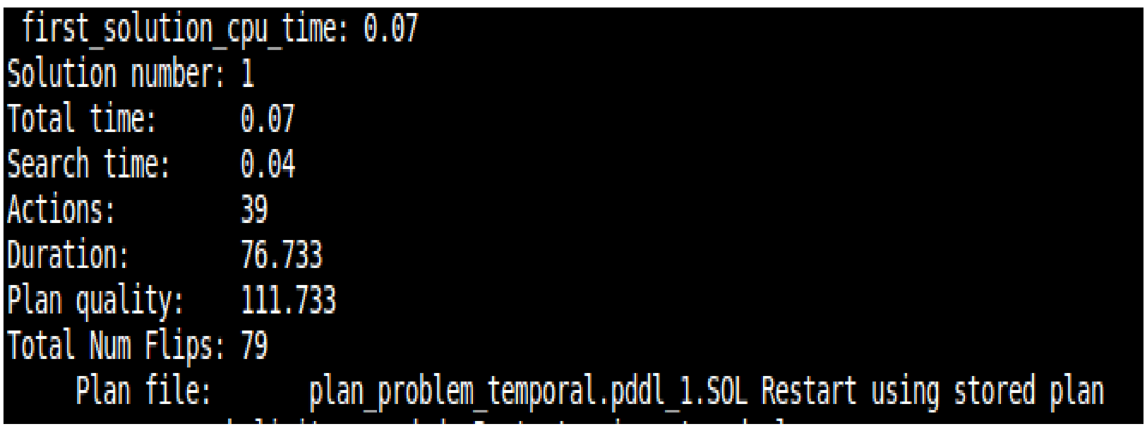
Fig-3: Output Plan

4.2 Experiment

We have experimented with our model in three different operating modes. Different operating modes give different results.

-n operating mode: -n operating mode gives us flexibility to choose the number of plans we want to generate. In our experiment, we considered generating 2 plans. And we get the following result:

./lpg-td -o domain_temporal.pddl -f problem_temporal.pddl -n 2



```
 first_solution_cpu_time: 0.07
Solution number: 1
Total time:      0.07
Search time:     0.04
Actions:         39
Duration:        76.733
Plan quality:    111.733
Total Num Flips: 79
     Plan file:       plan_problem_temporal.pddl_1.SOL Restart using stored plan
```

Fig- 4: First Solution

Generated Plan

```
METRIC_VALUE = 108.50
Solution number: 2
Total time:      0.25
Search time:     0.22
Actions:         36
Duration:        87.500
Plan quality:    108.500
Total Num Flips: 1146
     Plan file:       plan_problem_temporal.pddl_2.SOL
```

Fig-5: Second Solution

-Quality operating mode: This operating mode can give an optimal solution. It is mostly focused on quality and it ensures the best solution.

./lpg-td -o domain_temporal.pddl -f problem_temporal.pddl –quality

```
Solution found:
Total time:       6.01
Search time:      0.61
Actions:          25
Execution cost:   73.67
Duration:         61.000
Plan quality:     73.667
     Plan file:       plan_problem_temporal.pddl_1.SOL
```

Fig-6: Quality Mode Solution

- This mode can provide a quick but not optimal solution. It can add an unnecessary step to give a quick solution.

./lpg-td -o domain_temporal.pddl -f problem_temporal.pddl -speed

Generated Plan

```
METRIC_VALUE = 153.67
Solution found:
Total time:      1.37
Search time:     1.36
Actions:         43
Duration:        139.500
Plan quality:    153.667
Total Num Flips:  8344
     Plan file:       plan_problem_temporal.pddl_1.SOL
```

Fig-7: Speed Mode Solution

The figures show the results of our experiment. So, after experimenting with the model we designed, we can say that for optimal train schedule, the -quality operating mode is the best choice.

MACHINE CONFIGURATION

Operating System	70.04 LTS
Processor	Intel(R) Core (TM) i5-7200U CPU @ 2.50GHz 2.70 GHz
RAM	8.00 GB
SSD	240 GB
System	64-bit operating system, x64 – based processor

All resources and documentation are available on github. The Git repository of our work is: https://github.com/AminKaiser/Thesis-Temporal-Planning/tree/main/Version-2

QR Code of Repository

4.3 Result Analysis

We analyze the experimental results based on plan quality, duration, and actions. -n operation mode consumes most actions and duration and produces a lower quality plan.

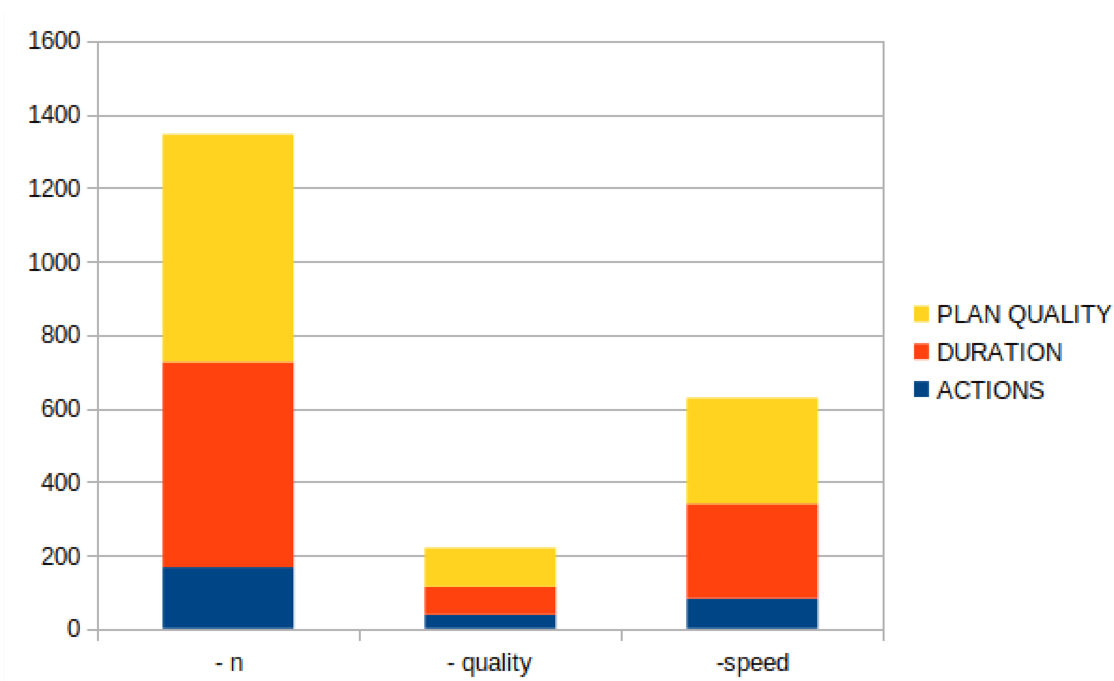


Fig-9: Comparison between operation modes

The -quality mode consumes shorter duration, and actions and produces the best plan quality. The -speed operation mode consumes medium-duration actions and produces medium-quality plans.

First, we have tested our design domain using the -n operation mode. We have generated two plans. But plans are not optimal. 167 actions, 558.367 duration, and 619.667 plan quality are found in this mode.

Then, we tested our design domain using the -quality operation mode. Plans are optimal. 38 actions, 75.800 duration, and 105.800 plan quality are found in this mode.

Lastly, we have tested our design domain using the -speed operation mode. Plans are not optimal. 82 actions, 257.800 duration and 387.800 plan quality were found in this mode.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In this work, a train scheduling model is presented which can optimize train scheduling problems like route choice, crossing, delays, and schedule disasters, etc. To develop domain models and problems, we use PDDL as a planning language and LPG-TD as an automated planner. We have tested our domain in three different modes. Each mode gives us different results.

-quality operating mode gives the best results. It optimizes the actions and duration, increases the plan quality, and gives the optimal solution.

5.2 Future Work

We considered the random velocity of a train in our problem. In the future, we want to add sensors to measure the speed of the train and GPS facilitation to collect the live coordinates of locations to work with real-time problems.

Also, this domain could be applied to bus or car route choosing and scheduling. This model might have the ability to produce the optimal plan for choosing the best route for the journey. This approach has the potential to play a key role in alleviating traffic congestion caused by the current high number of vehicles on the road.

Bibliography

[1] Wikipedia Bangladesh Railway. [Online]. Available: https://en.wikipedia.org/wiki/Bangladesh_Railway/ [Last visited on: 31 May 2022].

[2] Bangladesh Railway at a glance. [Online]. Available: https://railway.gov.bd [Last visited on: 31 May 2022].

[3] bdnews24 homepage. . [Online]. Available: https://bdnews24.com/bangladesh/2006/07/10/32-killed-40-injured-in-bus-train-collision-in-joypurhat-2nd-ld [Last visited on: 31 May 2022].

[4] Subarna Express rescued after two hours. . [Online]. Available: https://www.risingbd.com/sports-news/36933 [Last visited on: 31 May 2022].

[5] 2 die in Sirajganj train collision, 3 suspended. [Online]. Available: https://en.prothomalo.com/bangladesh/2-die-in-Sirajganj-train-collision-3-suspended. [Last visited on: 31 May 2022].

[6] Running train hits waiting Express 2 killed, 40 injured. [Online]. Available: https://thedailynewnation.com/news/9492/running-train-hits-waiting-express-2-killed-40-injured.html. [Last visited on: 31 May 2022].

[7] "10 killed as train hits microbus". The Daily Star. [Online]. Available: https://www.thedailystar.net/frontpage/news/10-killed-train-hits-microbus-1772011. [Last visited on: 31 May 2022].

[8] 16 killed as intercity train crashes into another in Brahmanbaria. [Online]. Available: https://www.thedailystar.net/country/news/12-killed-2-trains-collides-head-brahmanbaria-1826074. [Last visited on: 31 May 2022].

[9] Brahmanbaria train accident: Loco master of Turna Nishita admits negligence. Dhaka Tribune. [Online]. Available: https://archive.dhakatribune.com/bangladesh/nation/2019/11/14/brahmanbaria-train-accident-loco-master-of-turna-nishita-admits-negligence. [Last visited on: 31 May 2022].

[10] Accident halts Dhaka's rail link with northern districts after 1 killed. [Online]. Available:

https://en.prothomalo.com/bangladesh/accident/accident-halts-dhakas-rail-link-with-northern-districts-after-1-killed. [Last visited on: 31 May 2022].

[11] 3 die as train hits pickup van in Gazipur. [Online]. Available: https://en.prothomalo.com/bangladesh/local-news/3-die-as-train-hits-pickup-van-in-gazipur. [Last visited on: 31 May 2022].

[12] Eid holidaymakers face tailbacks, schedule disruptions [Online]. Available: https://www.tbsnews.net/bangladesh/eid-holidaymakers-face-tailbacks-schedule-disruptions-411382. [Last visited on: 31 May 2022].

[13] Tragedy of Northern passengers due to train schedule disruption [Online]. Available: https://dhakamail.com/national/15163?fbclid=IwAR3wbeUDPvu9Z-vHP4311ECsUkvs_F0PuOKFyv-ktmBSxuOilhrNhmbPKZ8. [Last visited on: 31 May 2022].

[14] Nir Lipovetzky, Daniele Magazzeni and Christian Muisc, An Introduction to the Planning Domain Definition Language. Patrik Haslum, Texas, 2019.

[15] Maria Fox and Derek Long, " pddl2.1 : An Extension to pddl for Expressing Temporal Planning Domains," Journal of Artificial Intelligence Research 20 (2003) 61-124, Dec 2003.

[16] What is a planner? [Online]. Available: https://planning.wiki/guide/whatis/planner. [Last visited on: 31 May 2022].

[17] LPG Homepage [Online]. Available: https://lpg.unibs.it/lpg/?fbclid=IwAR1QeJnUULYwbpUStrePKOYPqj2MeFUOZKJSfZierVr5HaZsLKA8ojk9xzE. [Last visited on: 31 May 2022].

[18] J. Scott Penberthy and Daniel S. Weld, “Temporal Planning with Continuous Change,” M. Sc. Engg. thesis, University of Washington, USA, Jul. 1994.

[20] Md. Moinul Islam, “Reviving the Railway for Sustainable and Safe Travel, A Study of the Dhaka- Narayanganj Railway Route,” M. Sc. Engg. thesis, Bangladesh University of Engineering & Technology (BUET), Dhaka, Bangladesh, May. 2013.

[21] Gabrio Curzio Caimi, “Algorithmic decision support for train scheduling in a large and highly utilised railway network,” Doctoral thesis, ETH Zurich, Switzerland, Sorengo, Switzerland, 2009.

[22] Alfonso Gerevini, Alessandro Saetti, Ivan Serina and Paolo Toninelli, " LPG-TD : a Fully Automated Planner for PDDL 2 . 2 Domains," 14th Int. Conference on Automated Planning and Scheduling (ICAPS-04) International Planning Competition abstracts, - 2004.