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**

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, signalling 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. [1]

[1] <https://railway.gov.bd>

**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. [8] |
| 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.[9] |
| 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. [10,11] |
| 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.[12] |
| 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. [13,14] |
| 07 November 2020 | An accident occurred in Kaliakoir, Gazipur: a woman was killed and five others injured. [15] |
| 21 May 2022 | An accident occurred at a level crossing in the Pubali area of Gazipur district: 3 people died in this accident. [16] |

[8] <https://bdnews24.com/bangladesh/2006/07/10/32-killed-40-injured-in-bus-train-collision-in-joypurhat-2nd-ld>

[9]  ["সুবর্ণ এক্সপ্রেস দুই ঘণ্টা পর উদ্ধার"](https://www.risingbd.com/sports-news/36933) [Subarna Express rescued after two hours]. [Risingbd.com](https://en.wikipedia.org/wiki/Risingbd.com) (in Bengali). 2014-02-05. Retrieved 2020-10-23.

[10] ["2 die in Sirajganj train collision, 3 suspended"](https://en.prothomalo.com/bangladesh/2-die-in-Sirajganj-train-collision-3-suspended). [Prothom Alo](https://en.wikipedia.org/wiki/Prothom_Alo). 2014-04-13. Retrieved 2020-11-11.

[11]["Running train hits waiting Express 2 killed, 40 injured"](http://thedailynewnation.com/news/9492/running-train-hits-waiting-express-2-killed-40-injured.html). The New Nation. 2014-04-14. Retrieved 2020 11-11.

[12]  ["10 killed as train hits microbus"](https://www.thedailystar.net/frontpage/news/10-killed-train-hits-microbus-1772011). [The Daily Star](https://en.wikipedia.org/wiki/The_Daily_Star_(Bangladesh)). 2019-07-16. Retrieved 2020-11-12.

[13]["16 killed as intercity train crashes into another in Brahmanbaria"](https://www.thedailystar.net/country/news/12-killed-2-trains-collides-head-brahmanbaria-1826074). [The Daily Star](https://en.wikipedia.org/wiki/The_Daily_Star_(Bangladesh)). 2019-11-13. Retrieved 2020-11-10.

[14] ["Brahmanbaria train accident: Loco master of Turna Nishita admits negligence"](https://www.dhakatribune.com/bangladesh/nation/2019/11/14/brahmanbaria-train-accident-loco-master-of-turna-nishita-admits-negligence). [Dhaka Tribune](https://en.wikipedia.org/wiki/Dhaka_Tribune). 2019-11-14. Retrieved 2020-11-10.

[15]https://en.prothomalo.com/bangladesh/accident/accident-halts-dhakas-rail-link-with-northern-districts-after-1-killed

[16]https://en.prothomalo.com/bangladesh/local-news/3-die-as-train-hits-pickup-van-in-gazipur

**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. [17] [18]

[17]<https://www.tbsnews.net/bangladesh/eid-holidaymakers-face-tailbacks-schedule-disruptions-411382>

[18]https://dhakamail.com/national/15163?fbclid=IwAR3wbeUDPvu9Z-vHP4311ECsUkvs\_F0PuOKFyv-ktmBSxuOilhrNhmbPKZ8

**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**

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. **[9 Book Chapter 1]** 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. [9] These are:

a. **Classical Planning:** Classical planning is the simplest planning, which assumes a deterministic, discrete, and essentially non-temporal world model.

b. **Numeric Planning:** Numeric planning relaxes the assumption of a discrete and finite model.

c. **Temporal Planning:** Temporal planning extends the classical planning problem with scheduling plan actions in time.

d. **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. **[9]**

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[9]:

\_ 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.[9]

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. [19]

[19 ] An Extension to pddl for Expressing Temporal Planning Domains by Maria Fox & Derek Long Department of Computer and Information Sciences, University of Strathclyde, Glasgow, UK

**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[2]. <https://docs.unity3d.com/Packages/com.unity.ai.planner@0.3/manual/index.html>

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. [2]

[2] <https://planning.wiki/guide/whatis/planner>

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.[2]

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[3]. [3] <https://lpg.unibs.it/lpg/?fbclid=IwAR1QeJnUULYwbpUStrePKOYPqj2MeFUOZKJSfZierVr5HaZsLKA8ojk9xzE>

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 [3]:

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.

**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.

**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.

**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.

**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.