CHAPTER 1

INTRODUCTION

Most research undertaken on timetabling problems to date is concerned with the scheduling of university courses. As traditionally defined, a timetabling problem is a general constrained assignment problem applicable to various areas of allocating jobs to machines in such a way that minimum costs are incurred or maximum returns are realized, or jobs are completed within the shortest time. Timetabling problems have numerous practical applications in many industries such as: aircraft, missile, communication, sport and recreation, computer technology, food, mining, petroleum and transport.

Over the years, constructing a timetable at a university has become more and more complex, mainly due to the growing number of students and courses under limited resources such as staff and lecture rooms [De Sousa *et al* (2007); De Palma & Lindsey (2001); Freling *et al* (2001)]. In the study of university timetabling problems several related terminologies have been used. These include: educational timetabling problems, class-teacher problems, course timetabling problems, student scheduling problems, classroom assignment problems and teacher assignment problems.

Application of timetabling problems in the area of transportation is the focus of our study. We focus particularly on the models and solution techniques that have been successfully applied to university course and examination timetabling to solve real world problems of taxi industry.

South Africa, on southern tip of the African continent, is bordered by the Atlantic Ocean on the west and by the Indian Ocean on the south and east. South Africa has nine provinces (see Figure 1.1), each with its own legislature, premier and executive council - and distinctive landscape, population, economy and climate. The provinces are the Eastern Cape, the Free State, Gauteng, KwaZulu Natal, Limpopo, Mpumalanga, the Northern Cape, North West, and the Western Cape.

Limpopo is South Africa's northernmost province, lying within the great curve of the Limpopo River. The province borders the countries of Botswana to the west, Zimbabwe to the north and Mozambique to the east. Limpopo is the gateway to the rest of Africa, with its shared borders making it favourably situated for economic cooperation with other parts of southern Africa. Each of the nine provinces of South Africa is further demarcated by district municipalities, and within each district are some local municipalities. Limpopo has five district municipalities and twenty-four local municipalities. The Capricorn district municipality is the busiest district in Limpopo since it houses the provincial government departments and main business establishments. Polokwane is the Central Business District (CBD) of Capricorn and the busiest city in the entire province.

Map

Limpopo
Gauteng
North West
Mpumalanga
Free State
KwaZulu Natal
Northern Cape
Eastern Cape
Western Cape

According to Statistics South Africa (Stats SA), in 2001 the Limpopo Province had a total of 4,995,535 people (Census 2001, Statistics South Africa). While the first democratic census of 1996 demarcated the province into six district municipalities, the cross-boarder re-demarcation of December 2005 reduced this number to five. The 4,995,535 people in 2001 indicated above have been re-adjusted to match the new demarcation. The population of Limpopo was estimated at 5,402,900 in 2007 (Mid-year population estimates 2007, Statistics South Africa).

ed population of the Province by district municipality and sex, in 2007. The results of the community survey of February 2007 by Stats SA have been employed to generate the data in Table 1.1.

Vhembe and Capricorn district municipalities have the highest population (23.7% each), while Waterberg has the least number of people (11.4%).

Table 1.1: Population of the Limpopo Province by district municipality and sex, 2007

District Municipality	Male	Female	Total	% Total
Mopani	496959	571620	1068579	20.4
Vhembe	564752	675295	1240047	23.7
Capricorn	572693	670449	1243142	23.7
Waterberg	291635	304455	596090	11.4
Greater Sekhukhune	492835	597590	1090425	20.8
Total Population	2418874	2819409	5238283	100.0

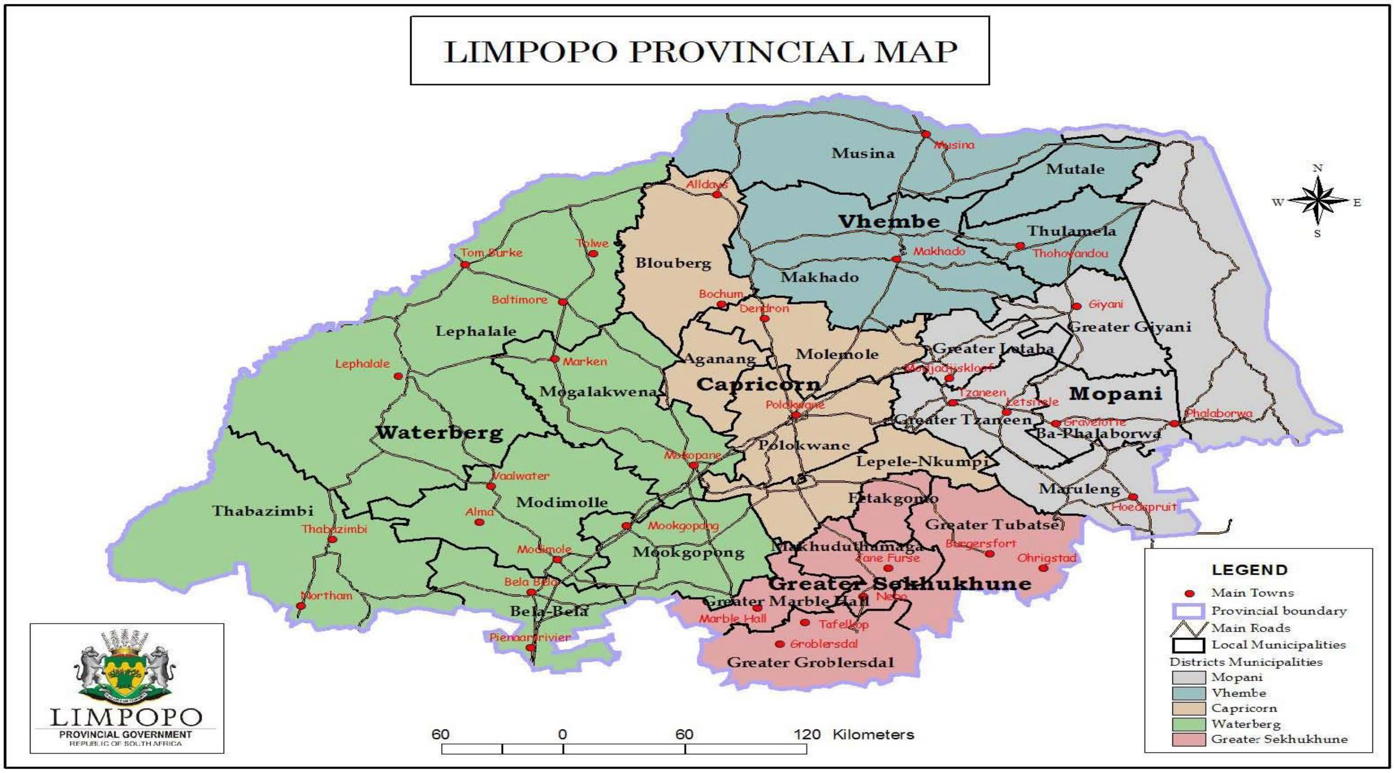
Source: Community Survey 2007, Statistics South Africa

The Capricorn District Municipality has five local municipalities. Greater Sekhukhune district also has five local municipalities; Mopani and Vhembe districts have four local municipalities each; and Waterberg has six local municipalities. The Provinceôs demarcation by district and local municipalities is displayed in Figure 1.2. Most commuters to the Capricorn district come from all districts and municipalities within the province, Gauteng, Mpumalanga and Zimbabwe. Booming mining industries and tourism attractions in Limpopo bring a lot of visitors to the province.

nd the University of Limpopo, Mankweng Hospital and new modern shopping complexes in Mankweng have led to many people traveling to and between these areas.

Figure 1.2: The Limpopo provincial map

About Limpopo - Location



http://www.limpopo.gov.za/about_otp/location.asp 3/11/2007 13:21:47

1.3 Problem statement

In South Africa, *taxis* are the most popular mode of transport and are used by the majority of the people. These taxis in the South African context are the minibuses that often carry 10-15 passengers. They are more popular e a shorter time to fill up with passengers, thereby reducing the waiting or idling period, although the latter are generally cheaper. They are also more popular than the cabs (the so-called 4+1ôs) because they are cheaper than the cabs.

Taxis in South Africa are privately owned by individuals, but usually the owners do not drive them. Year after year news headlines portray the industry as being at war resulting in deaths of drivers, taxi owners and commuters. Even though associations have been established to resolve these problems, little success has been achieved. Even more worrying about the taxi industry in South Africa is that many lives are lost as a result of lack of proper maintenance, driver fatigue due to long working hours, a rush to make quick bucks by overloading, and committing other road offences. Little do these taxi drivers realize the damage resulting from their wayward action.

The operational requirements for the taxi industry in South Africa are generally different from those of other modes of transport such as trains and buses. The latter follow scheduled arrival and departure times and stop only at predetermined stations or bus stops. Unlike trains and buses, taxis do not operate according to any specific pre-arranged schedule and could stop at any point along the route.

The main objective of taxis is to maximize profit, which they achieve by ensuring that all seats are occupied at each trip. This leads to passengers waiting long times for taxis during non-peak times. During busy times of waiting shorter periods, even though taxi drivers have a tendency to make an overload of passengers. At taxi ranks there are people called *Queue Marshals* who are responsible for securing positions for the taxis, controlling the whole process at the rank and ensuring that the daily schedule is obeyed. Taxi drivers have to book for the lines or positions every morning before they can start with their daily duties. With trains or buses, passengers usually buy tickets before embarking on a trip, but with taxis passengers pay after they are seated inside the taxi.

While it is acknowledged that the taxi industry plays an important role in the economy considering that the majority of South Africans depend on public transport, proper management and planning need to be put in place in order to reduce or prevent losses of lives.

Taxi timetabling includes drawing up a schedule to ensure that

(i) there is good management in terms of maintenance of vehicles, and proper time scheduling, and

(ii) there is no conflict between taxi associations and taxi owners, as well as among taxi drivers, while minimizing the number of road accidents.

Passengers often wait at taxi stops without having any idea of when the taxis will arrive. If there is a well-planned taxi schedule, passengers will know when to expect a taxi at their taxi stop. This will also benefit ho usually wait for long periods, and most often taxis pass them because there is no more room left, or else the taxis are overloaded with passengers. Proper taxi management and schedules can be helpful especially for large and busy taxi ranks anywhere in the country.

This research will particularly focus on two taxi ranks, one in the City of Polokwane and another in the Mankweng Township. These are the busiest ranks in Limpopo Province - with hundreds of passengers traveling on a daily basis from as early as 05H00 to as late as 23H00, depending on the season. The distance between these two ranks is about 30 km. Most passengers who travel between the two ranks are the communities around the City of Polokwane and Mankweng Township; the staff and students of the University of Limpopo ï Turfloop Campus (UL); and the staff of Mankweng Hospital that is about 1 km from the UL.

It is worth noting that South Africa, unlike most African countries, has a relatively good road infrastructure, with most networks having tarred roads. While Limpopo is considered as one of the poorest provinces in the country, the road link between Polokwane and Mankweng is a tarred road that is maintained at all times due to heavy traffic in the Capricorn district as indicated in Section 1.2 above. The national road (N1) that stretches from Cape Town all the way to Mussina in Limpopo with the main boarder with Zimbabwe, passes through Polokwane City.

It is believed that some of the reasons for the high rate of taxi accidents are the following:

(i) taxis are not well maintained;

(ii) taxi drivers do not abide by road traffic rules.

At present the taxi industry is run according to availability of passengers, hence passengers often suffer in their waiting. The aim of the project is to study and model a scheduling problem for the taxis, to minimize the waiting time of passengers, and in order to provide a better service to the public. The main objective of this study is to develop a model that would improve the service level of the taxi industry while minimizing costs incurred.

In tackling the problem of taxi industry, we use different optimization techniques that have been successfully applied in university timetabling problems to tackle the problem of taxi industry.

1.5 Preliminary data

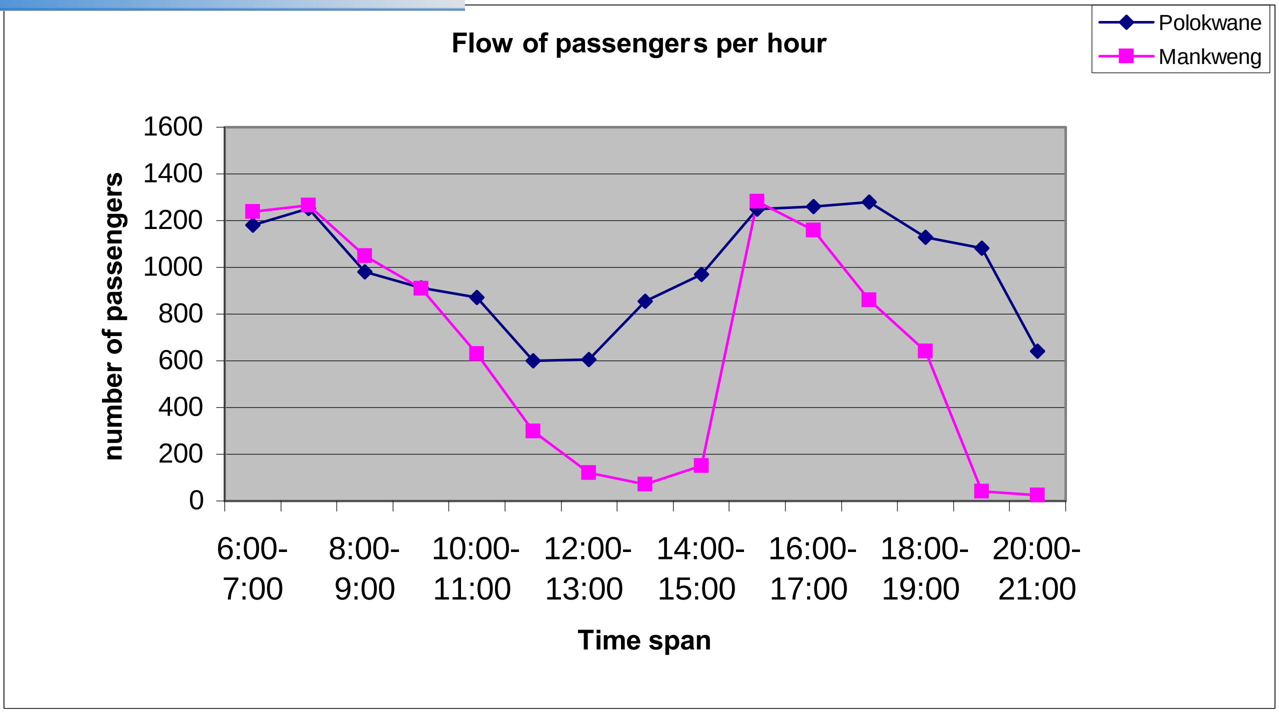
A structured questionnaire was developed to collect preliminary data from the Greater Mankweng Taxi Association (GMTA), an association that operates between the City of Polokwane and Mankweng Township. The aim of the questionnaire was to obtain information about the operations of taxis at the chosen locations, the number and sizes of taxis, operating hours, estimated number of passengers (during both peak and non peak hours), as d be useful when constructing the constraints in this research. Table 1.2, summarizes the information. A total of 505 taxis operate in the Polokwane and Mankweng ranks from 06H00 to 21H00. Each taxi makes three trips daily.

Table1.2: *Information about the operations of taxis in the Polokwane – Mankweng ranks*

Item	Polokwane	Mankweng
1. Number of taxis (505)		
1.1. During rush hours	250	256
1.2. During off peak	70	90
2. Starting time	06H00	06H00
3. Ending time	21H00	21H00
4. Floating taxis at rush hours	0	30
5. Number of round trips per taxi	3	3
6. Idling time		
6.1. During rush hours	3 minutes	3 minutes
6.2. During off peak	5 minutes	10 minutes

Figure 1.3 illustrates the flow of passengers on a daily basis at both ranks (Polokwane and Mankweng). During peak periods, particularly in the early morning hours (06H00 ï 09H00), and in the late afternoon hours (15H00 ï 18H00), the number of passengers is high at both ranks although the flow of passengers from Mankweng to Polokwane is relatively lower than from Polokwane to Mankweng in the afternoon rush hours. Around midday (10H00 ï 15H00), the flow of passengers from Mankweng to Polokwane is much lower than that from Polokwane to Mankweng.

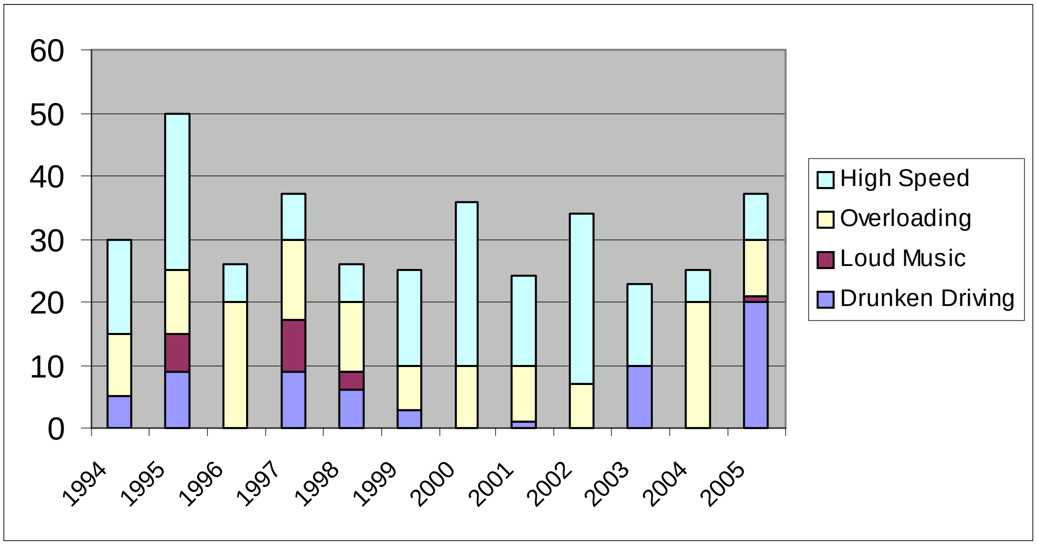
ssengers transported daily between Mankweng and



Information sought on the number of accidents since 1994 is displayed in Figure 1.4. Most accidents are caused by high speed driving, particularly in 1995, and from 1999 to 2003. However, drunken driving appears to be the major cause of accidents in 2005. Overloading was the main cause of accidents in 1996 and 2004, and loud music is featured mainly in 1995, 1997 and 1998. The information may have been duplicated since in some cases an accident may have been caused by a combination of factors. For instance, drunken driving and high speed could have been responsible for a single accident.

Given the findings above it is clear that there is need to put proper management and planning in place.

Figure 1.4: Number of accidents according to the factors



1.6 Organization of the report

The rest of the dissertation is organized as follows: In Chapter 2 we present a detailed literature review. In Chapter 3 we present the linear programming and integer programming models, and other models commonly used to solve timetabling problems. In Chapter 4 we present the computational analysis and propose the use of Genetic Algorithm. Chapter 5 gives conclusions.

CHAPTER 2

LITERATURE REVIEW

This chapter aims at reviewing timetabling problems in relation to the problem under study. Solution techniques and models applied to various types of timetabling problems with special interest to university (course and examination) timetabling problems are also discussed in this chapter. Section 2.2 provides a general definition and characteristics of a timetabling problem. Section 2.3 provides the main classes of a university timetabling problem studied in the timetabling literature. Section 2.3 examines the main types of public transport timetabling, and Section 2.4 provides the solution techniques applied to timetabling problems, in general.

2.2 Timetabling problem

According to Burke *et al* (2003), ñA general timetabling problem includes assigning a set of events (exam, courses, sports, matches, meetings, etc) to a limited number of timeslots while satisfying a set of constraintsò.

2.2.1 Hard and soft constraints

The constraints are categorized into two types, namely: hard and soft constraints.

Hard constraints are those that cannot be violated under any circumstances and must be rigidly fulfilled. In general, this means that:

turers, students, rooms, etc) can be assigned to different events at the same time.

(ii) Events (lectures) of the same academic period (same semester) must not be assigned to different resources at the same time.

The examples of the hard constraints:

(i) No student attends for more than one lecture at same time.

(ii) In the classroom, there must be enough space to accommodate all the students.

(iii) No lessons share one room at the same.

(iv) No lecturer can teach more than one class at same time.

(v) Any class cannot be taught by more than one lecturer at same time.

Soft constraints are desirable but not essential. Sometimes it is not possible to find solutions that satisfy all the soft constraints. Soft constraints can sometimes be violated and this fact is reflected in the value of the fitness function. The fitness function is a function that should be minimized in order to obtain the best solution. Since soft constraints are preferences only, more constraints can be imposed without any loss of generality of the problem. However, the imposition of a constraint will force a solution to ution may get altered by imposition of each soft constraint [Datta & Deb (2007)].

2.2.2 Feasible solution

Solutions are said to be feasible if they do not violate any of the hard constraints, while the violations of the soft constraints must be minimized as much as possible. The value of a feasible solution is the total number of soft constraint violations in the solution, and a solution with a low value means a good solution. Normally a cost is assigned to each type of constraints, with the hard constraints having higher associated costs than the soft constraints [Ranson & Ahmadi (2006)]. The core of the timetabling problem is to assign a suitable timeslot to each event such that all constraints are satisfied and the number of soft constraint violations is minimal. Burke *et al* (1995) define good feasible timetable as the timetables that are practical and with which the user is satisfied.

The general timetabling problems are known to be NP-hard or a class of hard-to-solve constrained optimization problems, due to a multiplicity of constraints that differ from one timetabling environment to another. There is no general model for timetabling problems because of the type of constraints. The timetabling problems are classified as constraint satisfaction problems. Hard constraints must be satisfied and these are modeled as the constraints of the problem and soft constraints are to be led as the objective function of the problem [Musli (2006)]. In some cases, the timetabling problem is formulated as a search problem, while in other cases the problem is formulated as an optimization problem. That is, what is required is a timetable that satisfies all the constraints and optimizes a given objective function that embeds the soft constraints [Schaerf (2007)]. Datta & Deb (2007) mention that though a class timetabling problem is tackled as an optimization problem, it does not have any fixed objective function to optimize. An objective function in this problem is just an arbitrary measure of the quality of a solution.

Figure 2.1 presents an extensible model or framework for general timetabling problems proposed by Ranson & Ahmadi (2006) based on the constraint satisfaction problem. Figure 2.1 is divided into two sections by a dotted line: an upper part illustrates the constraint satisfaction problem as the lowest level of the model built-up in layers. The whole idea has been inspired by an idea of ontology for constructing scheduling systems, which is structured around a constraint satisfaction model where activities are assigned resources to constraints. Table 2.1 provides the description of the classes found in the general timetabling model.

general timetabling model

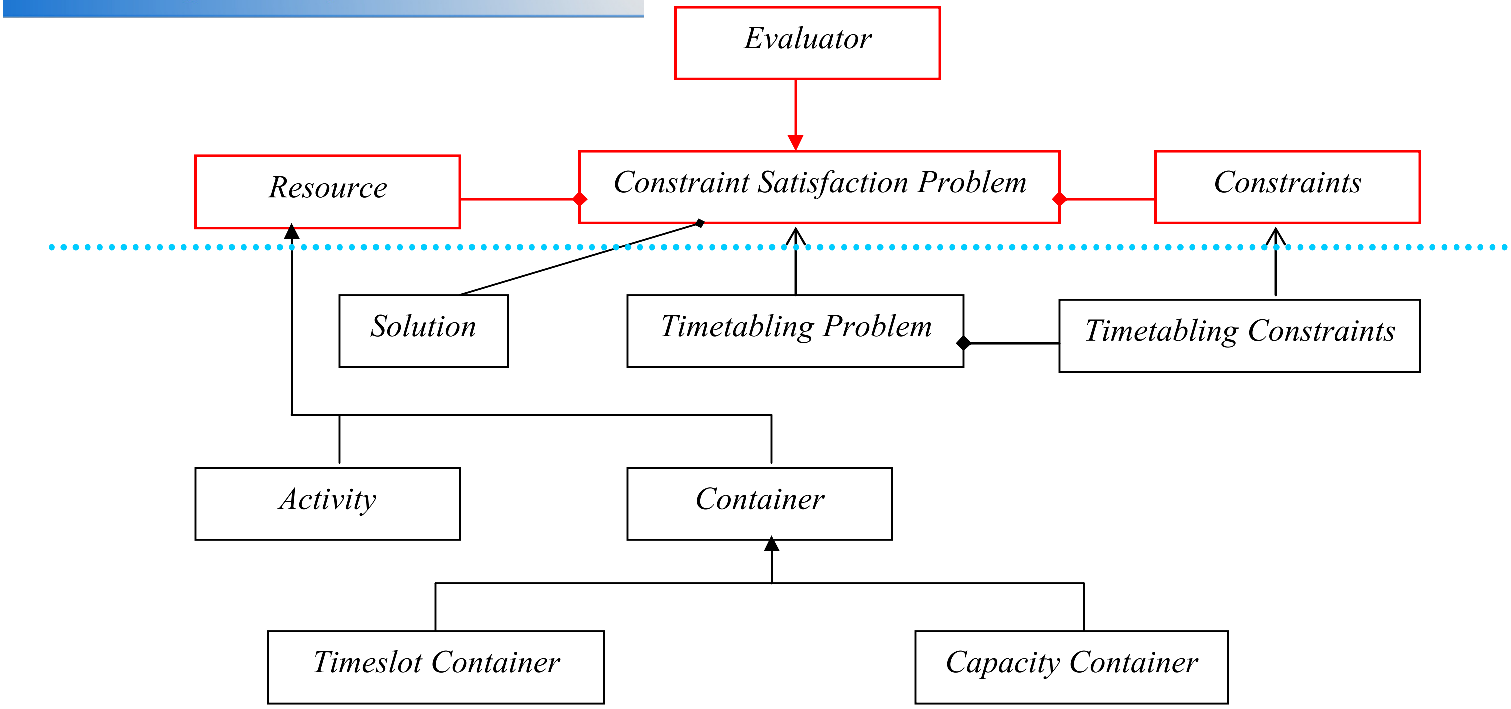


Table 2.1: Description of the classes found in the general timetabling Problem

Class	Description
1. Activity	Any activity that is to be timetabled
2. Timetabling Constraint	Constraints that access the timetabling resources
3. Container	A container where an activity can be timetabled
4. Capacity Container	A container with a limit to the number of resources that can be added
5. Timeslot Container	An ordered container with a specified duration

Timetabling problems arise in many real situations, including educational institutions (school and university timetabling), employment institute sports (timetabling of sports activities) and transport (train, airline and bus timetabling) etc. This study is focused on the application of university timetabling problems particularly on university course and examination timetabling to the taxi industry.

2.3 University timetabling problems

The university timetabling problem (UTP) is about finding an optimal distribution of classrooms and academic staff during a fixed period of time. Daskalaki & Birbas (2005) define a *university timetabling problem* ñas the process of assigning university courses to specific time periods throughout the five working days of the week and to specific classrooms suitable for the number of students and the needs of each courseò. The modelling of university timetabling problems depends on the objectives and constraints.

A UTP is a combinatorial optimization problem that has a huge number of possible solutions.

The two main classes of UTPs are the course timetabling problems (CTP) and the examination timetabling problems (ETP). These timetabling problems are related to each other but can be quite different. The main difference between these two is based on availability of rooms and capacity of the rooms. CTP requires that no two courses be scheduled in one room at the same time, while in ETP two or more examinations could be scheduled in one room. Often a student is not expected to write one another, hence examinations for the same set of students are not scheduled consecutively, but for course scheduling problems often double (or more) lecture periods are scheduled. For the purpose of our study both problems are considered.

2.3.1 University course timetabling problem

Chiarandini *et al* (2006) define the university course timetabling problem as an optimization problem in which a set of events (courses and lectures) has to be scheduled in timeslots and located in suitable rooms subject to a set of constraints. The objective is to find a feasible assignment that minimizes the number of soft constraint violations. The general CTP is known to be NP-hard, as are many of the sub-problems associated with additional constraints. However, several solution techniques have been employed, including graph colouring heuristics, Integer Programming, Tabu Search, Simulated Annealing, and Genetic Algorithm.

Although there are various formulations of the CTP which differ from each other mostly for the hard and soft constraints they consider, for the sake of generality, a basic version of the problem is described as follows:

A basic version of CTP can be generally described as the problem that consists of a set of n events, E, to be scheduled in a set of timeslots $T=\{t_1,t_2,\cdots,t_k\}$, a set of rooms R in which events can take place (rooms t of students S who attend the events, and a set of features F satisfied by events. Each student is already pre-assigned to a subset of events.

A feasible timetable is one in which all events have been assigned a timeslot and a room so that the following hard constraints are satisfied.

(i) No student attends more than one event at the same time.

(ii) Only one event is taking place in each room at a given time.

In addition, a feasible candidate timetable is penalized equally for each occurrence of the following:

(i) A student has a class in the last slot of the day.

(ii) A student has exactly one class during a day.

(iii) A student has more than two classes in a row.

Many constraints may be added as course timetabling problems vary from one institution to another. The above mentioned constraints are regarded as the common constraints in CTP. The objective is to minimize the number of soft constraints violations or maximize the number of soft constraints. Schaerf (2007) formulated the university course timetabling problem similar to the above problem with same description as optimization problem.

$$y_{ik} \qquad (i \; = 1,.....,q; \; k = 1,.....,p)$$

Subject to:

$$\sum_{k=1}^{p} y_{ik} = k_i \qquad (i = 1,....,q) \qquad (2.1)$$

$$\sum_{i=1}^{p} y_{ik} \leq l_k \qquad (k = 1,...., p) \qquad (2.2)$$

$$\sum_{i=\in S_i} y_{ik} \leq 1 \qquad (l = 1,....,r; \; k = 1,...., p) \qquad (2.3)$$

$$y_{ik} = 0 \text{ or } 1 \qquad (i \; = 1,.....,q; \; k = 1,.....,p) \qquad (2.4)$$

where $y_{ik} = 1$ if a lecture of course k_i is scheduled at period k, and $y_{ik} = 0$ otherwise. q is the number of courses, and p is the number of periods.

Constraints (2.1) impose that each course is composed of the correct number of lectures. Constraints (2.2) enforce that at each time there are no more lectures than rooms. Constraints (2.3) prevent conflicting lecturers to be scheduled at the same period. The objective function is given by

$$\text{Max} \quad \sum_{i=1}^{q} \sum_{k=1}^{p} d_{ij} y_{ik} \qquad (2.5)$$

where d_{ik} is the desiderability of having a lecture of course k_i at the period k.

can be fitted in either single objective optimization problem whereby the minimization of total soft constraint violations are taken as the only one objective or multiple objective optimization problems [Datta & Deb (2007)]. In order to fit soft constraint types in a single objective function, the notion of unit of penalty for each constraint and its weight should be defined [Gaspero & Schaerf (2007)]. Chiarandini *et al* (2006) consider a combined objective function for (i) a student has a class in the last timeslot of the day, (ii) a student has more than two consecutive classes, (iii) a student has to attend a single event on a day. Mushi (2006), in his objective function, included both hard and soft constraints, but higher penalties are assigned to hard constraints than soft constraints to discourage hard constraints from selection. The objective function is stated as follows:

Given a solution s and a set of k constraints,

$$\text{Minimize } f(s)=\sum_{i=1}^{k}\lambda_i f_i(s) \qquad (2.6)$$

Each function f_i represents one of the constraints and each λ_i is the weight given to constraint i depending on its importance.

timetabling problem

The university examination timetabling problem is one of the difficult combinatorial optimization problems that have been well studied by several researchers over the years. A lot of versions have been considered varying from institution to institution. Burke *et al* (1995) mention the following two constraints that are generally accepted to any timetabling problem and define the feasible timetable.

(i) No entity must be scheduled to be at more than one place at a time. In examination timetabling this would mean: no student can sit for more than one examination at any one time.

(ii) For each period in the resource demands made by the event scheduled for the period must not exceed the resource available.

Batenberg & Palenstijn (2007) define the ETP as the problem that consists of allocating a number of examinations in which students participate to timeslots in such a way that no student has two or more examinations in the same timeslots. The objective is to minimize the number of times a student has examinations in two consecutive timeslots, weighted by the time between the two consecutive slots. The formulation is as follows:

Suppose that there is a timetable T. Let d_{ij} be the weight for the time between examinations i and j in T, ($d_{ij} = 2$ if events i and j are on the onsecutive days, $d_{ij} = 0$ otherwise). Then the penalty for T is given by

$$T = \sum_{\substack{i \neq j \\ i,j}} c_{ij} d_{ij} + 5000 \times \text{ (number of unscheduled events)} \qquad (2.7)$$

For each pair of examinations (i, j), c_{ij} is the number of participants that they share. The problem is to minimize this penalty function over all timetables that satisfy the hard constraints.

Gaspero & Schaerf (2007) define the basic version of university examination timetabling problem as the problem of assigning examinations to timeslots by avoiding the overlapping of examinations having the students in common. The assignment is represented by a binary matrix $Y_{n \times p}$ such that $y_{ik} = 1$ if and only if the examination e_i is assigned to period k_j. The corresponding formulation is the following.

The objective is to find

$$y_{ik} \qquad (i \ = 1, \ldots\ldots, q; \ k = 1, \ldots\ldots, p)$$

Subject to:

$$\sum_{k=1}^{p} y_{ik} = 1 \qquad (i = 1, \ldots\ldots, q) \qquad (2.8)$$

$$\sum_{h=1}^{q} y_{ik} y_{jk} c_{ih} c_{jh} \leq 1 \qquad (k = 1, \ldots\ldots, p; \ \ i, j = 1, \ldots, n; \ \ i \neq j) \qquad (2.9)$$

$$y_{ik} = 0 \text{ or } 1 \qquad (i \ = 1, \ldots\ldots, q; \ k = 1, \ldots\ldots, p) \qquad (2.10)$$

where q is the number of examinations, and p is the number of periods.

n examination must be taken in exactly one timeslot. Constraints (2.9) state that no student takes two examinations scheduled at the same timeslot. From these constraints it is clear that one common hard constraint mentioned earlier that no events that are in conflict could be scheduled in the same timeslot, i.e. the examinations for the common students, should be separated.

The objective function is based on the soft constraints with different weights.

(i) *Second-order conflicts*: A student should not take two examinations in consecutive periods.

(ii) *Higher order conflicts:* A student should not take two examinations in consecutive periods at distance three, four or five.

(iii) *Preferences:* These can be given by lecturers and student for scheduling examinations to given periods.

Gaspero & Schaerf (2007) present a family of solution algorithms that are based on Tabu Search (TS) to a problem. However their results were not satisfactory in all instances after these TS-based algorithms have been compared with the existing literature in the problem, but they planned to improve their algorithms.

Soft constraint (i) second-order conflicts, is the most common type of soft constraints considered in the literature on examinations timetabling.

in the objective function the examinations belonging to the same group S_l scheduled at adjacent periods. The objective function is given as

$$\text{Minimize} \quad z = \sum_{k=1}^{p-1} \sum_{y=1}^{r} \sum_{i,j \in S_l} y_{ik} y_{ik+1} \qquad (2.11)$$

2.4 Public transport scheduling problem

The public transport scheduling problem is increasingly receiving attention in the literature. Several transport timetabling problems have been approached as vehicle scheduling problems, so timetabling and scheduling are synonyms in this paper. It is well known that timetabling can be seen as a form of scheduling, where the task is to allocate activities to available slots respecting some constraints [Sangheon (2004)]. The three classes of public transport timetabling problems that have been mostly given an attention by many researchers are: the train timetabling problem (TTP), the vehicle timetabling problem (VTP), and the airline timetabling problem (ATP).

The two sub-problems in VTP are: *vehicle and crew scheduling*. Traditionally, these two problems have been approached separately, so that vehicles are first assigned to trips, and in a second phase, crew are assigned to vehicle block (a set of two consecutive vehicle revenue trips to be operated, including the time taken to leave and return to the rank, where all vehicle trips originate and end) calculated before [Gintner *et al* (2006)].

hicle scheduling as the process of assigning vehicles to a set of predetermined trips with fixed starting and ending times, while minimizing capital and operating costs. The types of operational costs that might be involved are layover (idle time) and dead running (relocating the bus between locations with passengers, this includes leaving and returning to the depot). Different instances of vehicle scheduling problems have different types of objective functions. For example, to minimize the total sum of vehicle and crew costs such that both the vehicle and the crew schedules are feasible and mutually compatible [Wren & Wren (1995)]. In general, the objective function tends to be more complex in crew scheduling, it being a combination of fixed cost items, such as wages, and variable cost items, such as extra duty time. As a result, crew scheduling is usually harder to solve than vehicle scheduling [Sangheon (2004)].

A vehicle schedule is *feasible* if:

(iii) All trips are assigned to exactly one vehicle, and

(iv) Each trip is assigned to a vehicle from a depot that is allowed to drive this trip.

The vehicle scheduling is a step in the operational planning process in public transport. The starting point is a timetable, which defines the so-called timetabled trips. In any transport company, timetabling, vehicle scheduling, crew scheduling and crew rostering are the most important steps ss. Most of these steps are treated separately due to their inherent complexity. A typical example is the Bus Scheduling Problem (BSP) for which the operations planning and scheduling start off with the designing of a timetable of trips that have to be served by buses. Each trip has a starting time or location and destination time or location. BSP involves assigning a set of trips to a set of buses such that:

(i) The sequence of the trips for each bus is feasible: no trip precedes an earlier trip in the sequence.

(ii) Each trip is served by exactly one bus.

The crew scheduling problem consists in attribution to the drivers and collectors (crews), the job of driving the vehicles in such way that the trips of the different lines assisted by the company are executed with the smallest possible cost. This process of attributing tasks to the crews, also called a process of driver scheduling, is the construction of a group of legal shifts that cover all of the schedule blocks of a vehicle that is part of a large scale of vehicles reflecting all the operations of an organization. The crew scheduling problem is known as driver scheduling problem. Mauri & Lorena (2004) describe it as the one of formation of a matrix, where drivers appear in columns and tasks in rows. Each element $a_{ij} \in \{0,1\}$, $i \in M = \{1,...,m\}$ and $j \in N = \{1,...,n\}$, where m is the number of tasks (rows), and n the number of drivers (columns) of matrix A, and $a_{ij} = 1$ if the task i belong to j driverôs shift and $a_{ij} = 0$ otherwise

$$A = \begin{bmatrix} & & \ldots\ldots \rightarrow & a_{1n} \\ & & \ldots\ldots \rightarrow & a_{2n} \\ \downarrow & \downarrow & \ldots\ldots \rightarrow & \downarrow \\ a_{m1} & a_{m2} & \ldots\ldots \rightarrow & a_{mn} \end{bmatrix}$$

This matrix will be used to solve the following set partitioning problem (SPP).

$$\text{Minimize } z = \sum_{j=1}^{n} c_j x_j \qquad (2.12)$$

Subject to:

$$\sum_{j=1}^{n} a_{ij} x_j = 1 \qquad i = 1,\ldots\ldots,m \qquad (2.13)$$

$$x_{ij} \in \{0,1\} \qquad j = 1,\ldots\ldots,n \qquad (2.14)$$

where c_j is the cost of column j and $x_j = 1$ if column j belongs to the solution and $x_j = 0$ otherwise.

In bus crew scheduling problem (BCSP), generally the main aim for developing any bus crew schedule is to achieve optimum and dynamic schedule. Optimum schedules mean that resultant schedules should minimize the operational cost whilst dynamic means it is able to maintain such optimality throughout the schedule duration. However, the main obstacle of keeping such optimality throughout day-to-day operation is unpredictable events such as late crew or sick while on duty. Bus service unpredictable environment. Whenever an unpredictable event occurs, it affects bus operations even for other routes as well [Schaerf (2007)].

The BCSP is an extremely complex part of the operational planning process of transport companies. The planning process starts by the definition of vehicle schedules, aiming at minimizing the number of vehicles required. The constraints considered in the construction of a feasible BSCP solution can be divided into hard and soft constraints. The choice of the constraints to impose to a particular problem depends on the structure of the problem and on the operational planning rules of the company [De Sousa *et al* (2007)].

De Palma & Lindsey (2001) consider VTP that is defined as follows:

There is a given number of individuals who travel by transit on a single link. Preferred travel times and unit schedule delay costs for arriving early or late differ from person to person. Service on the route is provided by a fixed number of vehicles. Vehicle capacity constraints are ignored, so that a vehicle can carry any number of passengers with any congestion.

Sangheon (2004) considers the problem of routing vehicle through single depot or multiple depots. He defines the problem of vehicle scheduling problem as follows: Vehicle with a fixed capacity Q must deliver order quantity q_i $(i=1,...,n)$ of goods from a single depot $(i=0)$ to n customers. Knowing the distance d_{ij} between customers i and j $\left(i,j=0,...,n\right)$, the minimize the total distance traveled by the vehicles in such a way that only one vehicle handles the deliveries for a given customer and the total quantity of goods that a single vehicle delivers is not larger than Q.

Sangheon (2004) formulates this problem as follows: Let $G = (V, A)$ be a graph with a set V of vertices and a set A of arcs. $V = 0 \cup N$, where 0 corresponds to the depot and $N = 1,\ldots,n$ is the set of customers. For the set of arcs, $A = (\{0\} \times N) \cup I \cup (N \times \{0\})$, where $I \subseteq N \times N$ is the set of arcs connecting the customers, $\{0\} \times N$ contains the arcs from the depot to the customers, and $N \times \{0\}$ contains the arcs from the customer to the depot. Every customer $I \in N$ has a positive demand q_i. For each arc $(i, j) \in A$ there is a cost c_{ij}.

Furthermore, it is assumed that the vehicles are identical and have the capacity Q. All the above mentioned factors are assumed to be known in advance. The factors have the following variables: For each customer $i \subset N, y_i$ is the load of the vehicle when it arrives at the customer. Now the problem is to determine which of the arcs $(i, j) \in A$, are used by routes. For each arc $(i, j) \in A$, the decision variable $x_{ij} = 1$ if arc (i, j) is used by a vehicle, and $x_{ij} = 0$ otherwise. Formally

$$\text{Minimize} \quad z = \sum_{(i,j)\in A} c_{ij} x_{ij} \qquad (2.15)$$

$$\sum_{j \in V} x_{ij} = 1 \qquad \forall\ i \in N \qquad (2.16)$$

$$\sum_{j \in V} x_{ji} = 1 \qquad \forall\ i \in N \qquad (2.17)$$

$$x_{ij} = 1 \Rightarrow y_i - q_i = y_j \qquad \forall\ (i, j) \in I \qquad (2.18)$$

$$q_i \le y_i \le Q \qquad \forall\ i \in V \qquad (2.19)$$

$$x_{ij} \in \{0,1\} \qquad \forall\ (i, j) \in A \qquad (2.20)$$

The problem is to minimize the total cost that consist of travel costs and a fixed cost c of vehicles (included in the travel cost c_0 between depot and first customer). The objective is, first minimize the number of routes or vehicles, and then the total distance of all routes. By equations (2.16), (2.17) and (2.20), we require that every customer be visited exactly once. Equations (2.18) and (2.19) enforce the condition that the loads of the vehicle when arriving at the customer are feasible.

2.5 Solution techniques of timetabling problems

Over the past few years the university timetabling problem has been studied widely and several different algorithms have been developed to solve it. Local search approaches (Genetic Algorithm (GA), Simulated Annealing play an important role in the university timetabling literature. Local search is the common name for the group of methods that (on the whole) iteratively repeat the replacement of the current solution by a new one (Wren & Wren [1995]). Alvaraz-Valdes *et al* (2002) have developed the solution procedure based on TS.

Many authors have employed a hybrid method that consists of two or more phases or stages of the timetabling problem. Local search algorithms such as: GA, SA and TS appeared as effective algorithms for the second phase of the hybrid method. Liam *et al* (2002) presented a new hybrid algorithm for the examination-timetabling problem, with the following three stages:

(i) Constraint Programming: to obtain a feasible timetable.

(ii) Simulated Annealing: to improve the quality of the timetable.

(iii) Hill Climbing: for further refinement of the timetable.

SA is used to improve the quality of the timetable in the second phase. TS introduced by Glover (1988) turned out to be one of the most powerful tools for solving hard combinatorial problems.

Sangheon (2004) has combined GA and TS to develop Hybrid Meta-Heuristic methods to solve the Vehicle Scheduling Problem. Claessens *et al* (1998) present an algorithm that solved to optimality an allocation line problem by transforming its nonlinear integer programming formulation using programming methods. The algorithm is based upon constraint satisfaction and branch-and-bound procedures.

structed particularly for university class timetables. Nthangeni (1986) suggests an algorithm on the construction of university class timetable that is programmable on either the main frame or a microcomputer.

We review some techniques that have been frequently employed to solve various timetabling problems with the aim of designing an appropriate model for our own problem.

2.5.1 Graph colouring technique

De Werra (1985) employed the graph colouring technique in solving timetabling problems. The problem is presented as a network of arcs and nodes, and the solution involves finding a minimal set of colours such that no two adjacent nodes or edges have the same colour. While these techniques adapt well to small-scale problems, they fail to scale up to large ones. In a case where the problem is to schedule a course at the university with a fixed number of time slots, such a problem can be modelled as a graph colouring one. For large size problems, heuristics are needed in order to obtain appropriate solutions. Yanez & Ramirez (2003) introduced robust colouring that can be considered as an extension of a graph colouring problem. Scheduling problems can be modelled as Robust Colouring Problems (RCPs) so that the graph-based heuristics could be applied to solve them.

Daskalaki *et al* (2004) have modeled a UTP as an integer programming problem using 0-1 variables. The model provides constraints for a large number of different rules and regulations that exist in academic environments. More specifically, the model succeeds in creating timetables that are free from collision between courses and complete from all aspects. Moreover, it supports the scheduling of courses that require consecutive time periods as well as courses that require sessions that are repeated several times to accommodate different groups of students.

A major source of complexity in the complete Integer Programming (IP) model has been the constraints for consecutiveness. The relaxation is performed in a two-stage procedure. This solution approach suggests the relaxation of consecutive constraints during the first stage and their introduction during the second stage after all courses have been already assigned to days. Then the only concern is to re-arrange the courses of each day so that consecutiveness is achieved, whenever it is required.

2.5.3 Local search

A local search is a metaheuristic for solving computationally hard optimization problems. It can be used on problems that can be formulated as finding a solution maximizing a criterion among a number of candidate solutions. Local search techniques are non-exhaustive in the sense that they asible (or optimal) solution, but they search non-systematically until a specific stop criterion is satisfied.

The heuristics are generally employed for two different purposes

(i) They can be used within the context of an exact optimization algorithm to speed up the process of reaching the optimum.

(ii) They are simply used to find a ñgoodò solution to the problem. The resulting solution is not guaranteed to be optimum and, in fact, its quality relative to the true optimum may be difficult to measure [Taha (1989].

Consider an optimization problem, and let S be a possible search for it. Function N, which depends on the structure of the specific problem, assigns to each feasible solution $s \in S$ its neighbourhood $N(s) \subseteq S$. Each solution $s' \in N(s)$ is called a neighbour of s [Schaerf (2007)].

A move is the operation of taking one event and move it to one of its possible places P and the move neighbourhood $N(s)$ of s consists (is a set) of all solutions s' obtainable from s in one move. The move value is the change in a solution value when moving from s to s'. Thus a move equals to -1 means that the move improves the solution slightly [Arntzen (2007)].

Genetic algorithm (GA) is a heuristic search used to find approximate to difficult-to-solve problems through application of the principles of evolutionary biology (Chromosomes reproduction) and Computer Science. GA uses biologically derived techniques such as: inheritance, mutation, nature selection and recombination (or cross-over). The two main operators, namely crossover and mutation, transform timetables to form the next generation with better timetableôs quality by improving the initial population of a feasible timetable.

The crossover operator is one of the most essential genetic operators. Its task is the realization of the deterministic search and it tries to advance in the problem space by applying the existing knowledge [Gyori *et al* (2001)]. The mutation operator introduces random modifications. It alone induces a random walk through the search space. The role of the mutation operator in the GA is the assurance of the heuristic search. It tries to get to the individuals found in the undiscovered part of the problem space.

GA is used to develop some approaches to create new schedules from other existing ones (a population of individuals is created randomly). The concept is to create a fitness function that determines the quality of a schedule and compares the newly created schedule with the previous one. Each schedule is evaluated according to a set of criteria that are included in the fitness function, for example, the length of a schedule, how many students have to row and the spare capacity in each of the rooms.

Only schedules that get a higher ranking from the fitness function will be followed on. Although this approach is found to be powerful, it only works on a convex solution space.

The basic algorithm is given as follows:

```
1.  Randomly initialize population (t);
2.  Determine fitness of population (t);
3.  Repeat the following:
        •  Select parents from population (t);
        •  Perform crossover on parents creativity population (t+1);
        •  Perform mutation of population (t+1);
        •  Determine fitness of population (t+1);
    until the best individual is good enough.
```

A population of individuals is maintained within the search space for a GA, each representing a possible solution to a given problem.

2.5.5 Simulated annealing

Simulated Annealing (SA) is a search technique that tries to avoid becoming trapped in a local optimum by allowing some uphill steps or moves. This is done as follows: it solves the timetabling problems by allowing a worse e considered at times. If an improved local search move is better than its current position then it is always accepted. If the move is worse (i.e. lesser quality) then it will be accepted based on some probability which depends on the relative deterioration in the evaluation function value, such that the worse a move is, the less likely it is to accept it. Burke *et al* (1995), argue that SA is similar to Hill-Climbing but accepts a worse solution with a probability distribution known as the Metropolis distribution, defined as follows:

$$p_{\text{accept}}\ (T,s,s') = \begin{cases} 1 & \text{if} \quad f(s') < f(s) \\ \exp\left(-\dfrac{f(s') - f(s)}{T}\right) & \text{otherwise} \end{cases} \qquad (2.21)$$

where s is the current solution, s' is a neighbour solution and $f(s)$ is the evaluation function. The temperature parameter T, which controls the acceptance probability is allowed to vary over the course of the search process.

2.5.6 Tabu search

Tabu search (TS) is a local search metaheuristic that relies on specialized memory structures to avoid entrapment in local optimum and achieve an effective balance of intensification and diversification. *Intensification* strategies involve changing the choice rules to intensify the search to examine neighbours of elite solutions. The idea is that if certain regions the past they may possibly yield better solutions in the future. The *Diversification* stage encourages the search process to examine unvisited regions and to generate solutions that differ significantly.

More precisely, TS allows the search to explore solutions that do not decrease the objective function value only in those solutions that are not forbidden.

In general, TS starts with an initial solution that has been previously constructed, and runs through an iterative process by means of which it seeks to improve the objective function value of subsequent solutions [Malachy & Sinead (2003)]. This is usually obtained by keeping track of the last solutions in terms of the move used to transform one solution to the next. When a move is performed it is considered to be tabu forbidden for the next N iterations, where N is the tabu status length. A solution is forbidden if it is obtained by applying a tabu move to the current solution.

At each iteration of the search, a neighbourhood is examined to construct a new solution [Gunadhi *et al* (1996)]. According to the given neighbourhood, a move for the timetabling problem is defined by moving one event or by swapping two events. A move is forbidden if at least one of the events involved has been moved less than N steps before.

ublic transport scheduling

Vehicle-scheduling problems are optimization problems that have been categorized as NP-hard. (An *NP-hard problem* is any optimization problem that cannot be solved in polynomial time). Such problems have therefore been approached using heuristics and meta-heuristics due to the complexity of applying exact techniques to global optimality.

Heuristic methods have achieved good solutions to real-time transport (vehicle and train) scheduling problems. Ebben *et al* (2005) present heuristics for the dynamic vehicle-scheduling problem with multiple resource capacity constraints. A serial scheduling method (a priority rule based on scheduling heuristic) consists of two elements: a schedule generation scheme and a priority rule. A *feasible schedule* is generated by extending a partial schedule. At each stage, an activity that is not yet scheduled is selected according to the specified priority rule. Each activity can only be scheduled once. Malachy & Sinead (2003) used heuristics analogous to the methods employed in existing train planner manual methods for many advantages that include the structure of the problem to reduce the search space. Sangheon (2004) developed a Hybrid Meta-Heuristic Method for the vehicle scheduling problem. A solution procedure is based on a GA and TS. TS is used to generate the seeds of a GA heuristic. Wren & Wren (1995) provide an example of GA for public transport driver scheduling.

onsider two location models, namely: the line model and the circle model to find an optimal solution for a given number of public transport vehicles on a single transit line. In their problem, firstly, it is assumed that the travelersô desired travel times are distributed over a segment of the day and rescheduling of trips between days is impossible. Secondly, it is assumed that desired travel times are distributed around the clock and that rescheduling of trips between days is possible.

CHAPTER 3

SCHEDULING MODELS: AN APPLICATION

APPLICATION OF UNIVERSITY TIMETABLING PORBLEM

In this chapter we present an application of university (courses and examinations) timetabling models and mathematical formulation described in the literature review (Chapter 2) to the taxi-timetabling problem. In section 3.2 the application of university timetabling problem is presented and section 3.3 gives the mathematical formulation models to model our problem.

3.2 Application of university timetabling problem

In this section, we describe how the application of university timetabling is employed to the taxi-timetabling problem.

3.2.1 Problem situation

We consider a single depot vehicle-scheduling problem for the Greater Mankweng Taxi Association (GMTA) that operates between the City of Polokwane and Mankweng Township. We also consider one working week schedule of all trips starting from 05H00 to 20H00 to the set of taxis. The ordinary daily duty consists of more than 1500 trips, from both two ranks and served by over 500 taxis, except Saturday and Sunday where there will be lesser trips.

Since an association operates from two ranks (control points), the final timetable should incorporate all the timetables from both ranks. For an example, the taxis will have to be assigned to trips at both ranks.

$T = \{t_1, t_2, \ldots, t_n\}$ be a finite set of taxis.

$t_i(A)$ and $t_i(B)$ be a taxi t_i at rank A and B respectively, $i = 1, 2, \cdots, n$. Every taxi is starting from and returning to the same rank.

$\mathrm{Dep}(t_i, A)$ and $\mathrm{Dep}(t_i, B)$ denote the departure times for taxi t_i at rank A and rank B respectively.

$\mathrm{Arr}(t_i, A)$ and $\mathrm{Arr}(t_i, B)$ denote the arrival times for taxi t_i at rank A and rank B respectively. A taxi trip of taxi t_i is described by an ordered pair $\{\mathrm{Dep}(t_i, A), \mathrm{Arr}(t_i, B)\}$. A taxi block of t_i is given by a set of trips as,

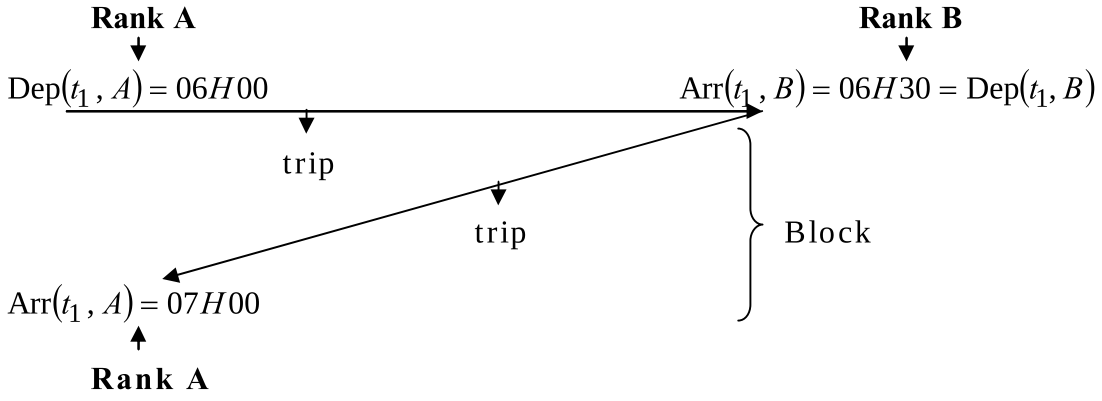
Block $= \{(\mathrm{Dep}(t_i, A), \mathrm{Arr}(t_i, B)), (\mathrm{Dep}(t_i, B), \mathrm{Arr}(t_i, A))\}$.

Figure 3.1 illustrates the first taxi $t_1(A)$ that starts the day at 06H00 from Rank A to complete a trip to Rank B at 06H30 and to complete block back to Rank A, now at 07H00. This implies that:

$$\mathrm{Dep}(t_1, A) = 06H00$$

$$\mathrm{Arr}(t_1, B) = 06H30 = \mathrm{Dep}(t_1, B)$$

$$\mathrm{Arr}(t_1, A) = 07H00.$$

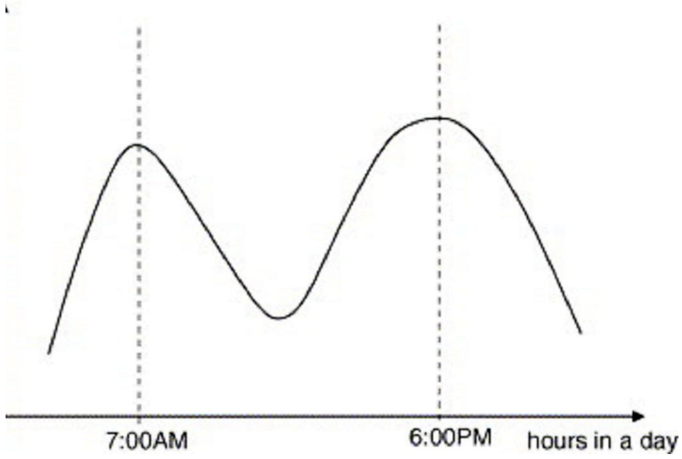


Due to the fact that in GMTA individuals privately own taxis, every owner is responsible for a driver. Crew (driver) scheduling as a sub-problem of VSP is considered in a different way from a case whereby one company owns all the vehicles. In many public transportation companies a crew may leave a vehicle and another takes over and the vehicle continues in service, unlike in taxi industry a relief of driver is a relief on a vehicle. In vehicle scheduling, the problem is to construct blocks of consecutive trips. Each block must start and end at a depot, while satisfying appropriate operational restriction.

3.2.2 Taxi scheduling

In general the VSP consists of assigning vehicles originating and terminating from a central depot to time tabled trips in such a way that each trip is carried by one vehicle, a set of constraints is satisfied, and a cost function is minimized. The timetabling problem for taxi (minibus) transport involves the scheduling of several taxis from origin taxi rank to transport and return to the origin rank. The problem data are a set of trips, each one defined by the starting and ending times and locations. The timetable is built in function of passenger demand distribution along a day. The passenger flow is the one of the most sensitive factors in the transportation scheduling problem to be looked at. The analysis of the historical passenger flow can be used to predetermine the pattern of peak and off-peak duration everyday. Figure 3.2 illustrates passenger flow during peak and off-peak hours along a day.

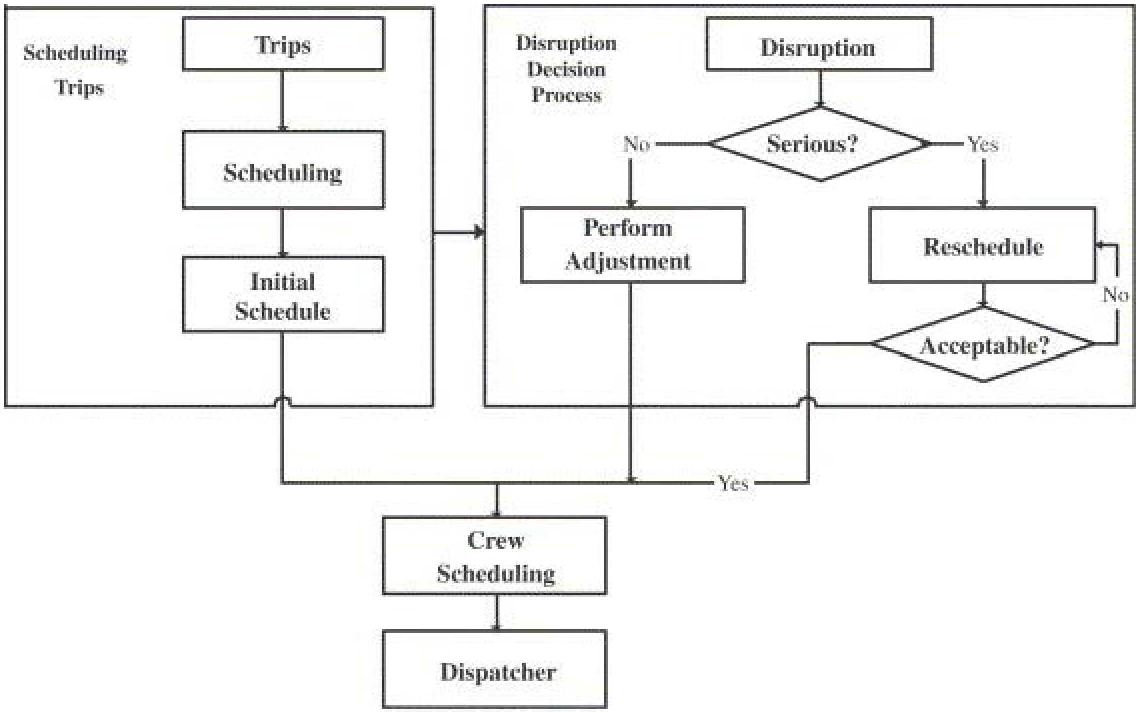
Figure 3.2: Estimated number of passengers transported daily



Source: Rodrigues et al (2007)

Another sensitive factor of the transportation scheduling problem is disruptions of a schedule. Transportation systems often encounter disruptions that prevent them from operating as planned. Severe weather conditions, accidents, and the breakdown of vehicles are examples of and the rescheduling of vehicles. In case of disruptions, a schedule can also be adjusted depending on the nature of disruption. If it is a serious disruption, it needs to be rescheduled to complete all the remaining trips that include disrupted one. Figure 3.3 describes a typical process involving both the scheduling and the rescheduling of disrupted timetable. Chang and Chung (2005) solve the rescheduling timetable problem by using objective function as the difference between the original timetable and re-scheduled timetable.

Figure 3.3: Scheduling and rescheduling of timetable.



Source: Chang and Chung (2005)

3.2.3 The hard constraints

For this study, the following are considered to be hard constraints and must be satisfied:

an be scheduled for more than one trip at the same timeslot, and no passenger can be assigned for more than one trip in the same timeslot.

(ii) Each taxi can only take one trip at the same time.

(iii) The capacity of a taxi should not be exceeded (i.e. there should be no overloading).

(iv) Each block of trips starts and ends at the same depot.

(v) Each taxi should complete its block without any interruption. In order to do so it must be assigned two consecutive trips in different ranks.

(vi) Each taxi must not be scheduled more than once before all taxis from a pool are scheduled once.

(vii) Only roadworthy taxis can be scheduled for a trip.

3.2.4 The soft constraints

A timetable that satisfies the hard constraints discussed above is called a feasible timetable. However, a feasible timetable does not mean a good timetable for business concerns. There are also a number of additional constraints that should be satisfied whenever possible in order to make our timetable more practical in the real world. If any of these constraints is to the solution. The soft constraints are as follows:

(i) The consecutive trip departure times in each hour must be as uniformly spaced as possible, with a typical tolerance of two minutes.

(ii) The number of trips, in each hour and direction, must be enough to cover the passenger demand.

(iii) No trip with fewer passengers should be assigned.

(iv) A driver should not have two blocks in adjacent timeslots.

(v) Every driver has a minimum and a maximum limit of weekly work-hours.

3.2.5 The objective function

Since many taxi drivers and owners have hard constraints, e.g. they may need to maximize profits. The objective function in our problem is to improve the timetable with respect to passengersô need. The objective of this problem is to minimize the waiting of passengers and provide the better service to the public. The first two soft constraints ((i) and (ii)) discussed above address the waiting times of passengers. If the departure times are not uniformly spaced, in some cases the length of waiting time will be longer or shorter. However, the waiting times might not be the same. The length of waiting times depends on the predicted volume of passenger flow.

o considered is that, from the taxi industryôs point of view, it is preferable to maximize profit by ensuring that all seats are occupied at each trip, implying that if the passengers are fewer than the capacity of the taxi, they (passengers) will have to wait until the taxi is fully occupied. It becomes the problem of minimization of waiting times of passengers against maximization of profit. A solution for this problem should minimize the number of trips with fewer passengers. However, in some occasions the minimization of these trips may not be possible as it depends on the volume of passengers.

3.2.6 Evaluation of the feasible Schedule

In our problem, a feasible schedule would be the assignment of a set of trips to a set of taxis within the limited number of timeslots such that: the passengers should not wait for long time at stations. To evaluate whether or not a schedule is feasible one could consider several factors, but in our problem we consider a set of soft constraints that we will try to avoid violating. A cost function that computes a penalty for any feasible schedule is derived based on the number of soft constraints violated.

The objective function in this problem is constructed by minimizing the number of soft constraint violations. If the total resulting from violating the soft constraints can be minimized so as to approach zero, the generated timetable will be close to the optimal solution. An infeasible assignment is considered useless and is therefore discarded [Muller (2002)].

ming model

In this section, we present integer and linear programming models to model our problem.

The formal mathematical definition of the taxi-timetabling problem we deal with in this study is as follows:

The taxi-timetabling problem (TTP) aims at scheduling a number of events (trips) to the available timeslots, and suitable taxis while satisfying a set of constraints. For the constraints of this problem we consider the problem instances taken from literature [Burke *et al* (2006), Daskalaki *et al* (2004), Gunadhi *et al* (1996)]. Given a set of n, events E to be scheduled in a set of timeslots, a set of rooms R in which events can take place (rooms are of a certain capacity), a set of students (passengers) S who attend the events, and a set of features F satisfied by events.

3.3.1 Assumptions

The following assumptions have been made

(i) The taxis are identical in terms of capacity.

(ii) In a course or examination timetabling problem, there are a number of enrolled students for that course, while in taxi timetabling the number of passengers is not fixed. So it is assumed that the number of passengers to be transported per trip is determined by the vehicle capacity.

d the distance are known.

(iv) The consecutive trip departure times at each rank are uniformly spaced. The spacing is divided into peak and non-peak hours, with asuumption for peak hours it is assumed a typical tolerance of two minutes and non-peak with 10 minutes. Peak hours are between 6H00 and 9H00 and 16H00 and 19H00.

(v) All trips start and end at the two ranks, there are assigned around floating taxis collecting passengers to the ranks.

(vi) A taxi makes two consecutive trips without an interruption from different ranks, thus makes a complete block. (See Figure 3.1).

3.3.2 Definitions

For any vehicle scheduling there are terms that are considered in the literature, but in this study we consider some of them:

(i) *Trip*: a one-way movement of a vehicle between two terminuses or ranks.

(ii) *Block*: a set of two consecutive vehicle revenue trips to be operated, including the time taken to leave and return to the rank, where all vehicle trips originate and end.

(iii) *Layover arc:* a length between the end of a trip and the start of a (later) trip at the same rank.

er of vehicles in the association.

(v) A floating trip is a trip that a vehicle is moving to or from the depot (possibly moving without passengers).

3.3.3 Notations

Parameters	*Description*
n	The number of trips or timeslots per day that start at rank A
m	The number of trips or timeslots per day that start at rank B
p	The total number of taxis in the association
T_i	A trip i where $i \in \{1, \cdots, n)$
T_j	A trip j where $j \in \{1, \cdots, m\}$
L_B	An arrival time at rank B of taxi i minus a starting time of daily operation
w	A time between timeslots at rank A
$d_{i,i+1}$	A penalty for a duration between trip i and trip $i+1$
$d_{j,j+1}$	A penalty for a duration between trip j and trip $j+1$
δ_i	A tolerance time for dwelling at rank A
δ_j	A tolerance time for dwelling at rank B

th n trips or timeslots at rank A, and m trips or timeslot at rank B and a set of p taxis. Since we attempt to assign number of trips to the set of taxis at two ranks, then the variable definition is in two separate decision variables.

$$x_{ik} = \begin{cases} 1 & \text{If trip } i \text{ or timeslot } i \text{ is assigned to taxi } k \text{ at Rank A} \\ 0 & \text{Otherwise} \end{cases}$$

$$x_{jk} = \begin{cases} 1 & \text{If trip } j \text{ or timeslot } j \text{ is assigned to taxi } k \text{ at Rank B} \\ 0 & \text{Otherwise} \end{cases}$$

A set of hard constraints is generated as follows

(i) Each trip or timeslot must be assigned to one taxi.

$$\sum_{k=1}^{p} x_{ik} = 1 \qquad i = 1,\ldots\ldots,n, \quad i \in A \tag{3.1}$$

$$\sum_{k=1}^{p} x_{jk} = 1 \qquad j = \left(\frac{L_B}{w} + 1\right),\ldots\ldots,m, \quad i \in B \tag{3.2}$$

$$\sum_{k=1}^{p} x_{ik} - \sum_{k=1}^{p} x_{jk} = 0 \qquad i = 1,\ldots\ldots,n, \;\; i \in A, \quad j = \left(\frac{L_B}{w} + 1\right),\ldots,m \;\, j \in B \tag{3.3}$$

mpatible with a trip, i.e. the number of passengers assigned to a trip must not exceed each taxi capacity.

$$\sum_{i \in R} x_{ik} = 0 \qquad k = 1,\ldots\ldots, p \qquad (3.3)$$

$$\sum_{j \in R} x_{jk} = 0 \qquad k = 1,\ldots\ldots, p \qquad (3.4)$$

where R is the set of taxis whose capacity is exceeded by the number of passengers.

(iii) Each taxi can fit to one trip.

$$\sum_{i=1} x_{ik} \leq 1 \qquad \mathrm{k} = 1,\ldots\ldots, p \qquad (3.5)$$

It is still possible that large and small number of passengers than the taxiôs capacity get assigned to a taxi. To avoid such scenario, and distribute the trips fairly and efficiently among the available taxi, we add the following optimization objective function:

$$\text{Minimize z} = \sum_{i=1}^{n} \sum_{k=1}^{p} c_{ik} x_{ik} + \sum_{j=1}^{m} \sum_{k=1}^{p} c_{jk} x_{jk}$$

where $c_{ik} = 5 \times \delta_i$ and $c_{jk} = 5 \times \delta_j$ are the unit costs (tolerance dwelling time + penalty) from trip i to trip $i+1$, and from trip j to trip $j+1$ respectively. It e for dwelling is 5. Tolerances are defined to control the minimum and maximum spacing between consecutive trip departures. The penalties are defined by

$$\delta_i = \begin{cases} 2 & if \;\; d_{i,i+1} \le 5 \\ 1 & if \;\; d_{i,i+1} > 5 \end{cases} \qquad \delta_j = \begin{cases} 2 & if \;\; d_{j,j+1} \le 5 \\ 1 & if \;\; d_{j,j+1} > 5 \end{cases}$$

CHAPTER 4

ANALYSIS AND RESULTS

ents and results

For validation of the proposed algorithms, we perform tests on randomly formed instances based on the real world problem of taxi industry in South Africa. The parameter values for Genetic Algorithm Annealing are shown in Table 4.1. The number of trips on instances tested is 30 and 50.

Table 4.1: Genetic Algorithm parameters.

Number of trips	**30**	**50**
Population size	30	30
Maximum generations	30	60
Crossover type	2-point	2-point
Crossover probability	0.7	0.7
Mutation type	Design Wise	Design Wise
Mutation probability	0.2	0.2

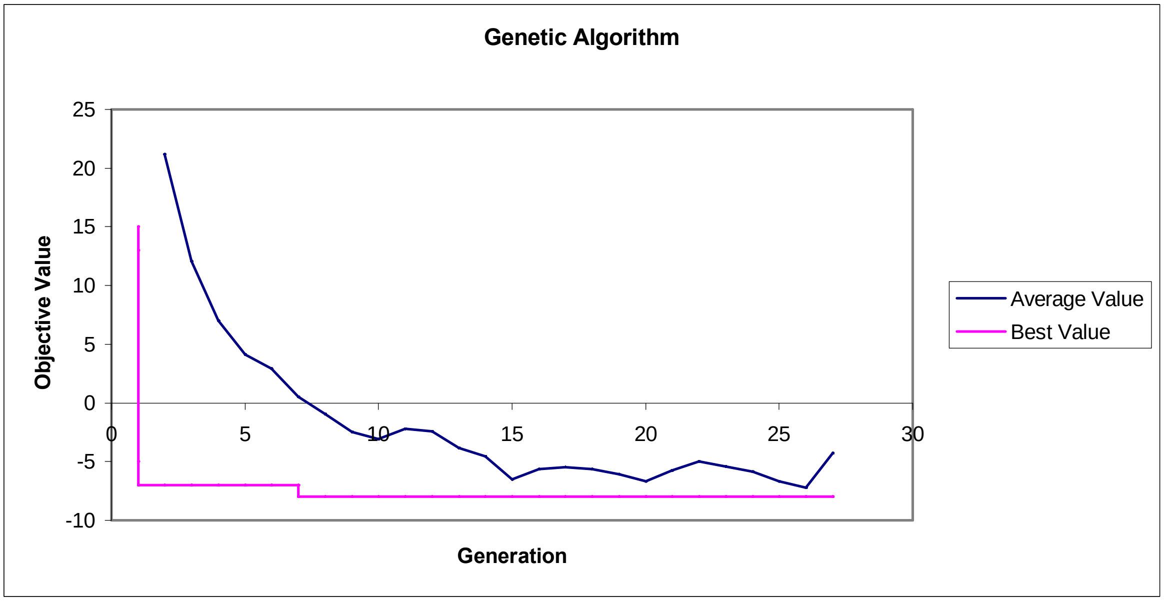
The computational results are shown in Table 4.2. Our algorithms are implemented in Optworks Optimization tool with Microsoft Excel.

Figures 4.1 and 4.2, show the average value and best value in our test, on instances of 30 and 50 trips, respectively. In both figures 4.1 and 4.2, a y-axis is the objective function value. Figure 4.1 shows that in the initial stage of the GA process, the best value of objective function is 15, after 2 generations it decrease to ï 8 as a net objective value, but the best of objective is 8. Tables 4.2 and 4.3, show that the number of violated constraints is zero for both instances of 30 and 50 trips.

utput for instance of 30 trips

OptWorks Optimizer Outputs		
Algorithm	Genetic Algorithm	Value
Best Net Objective Value	-8	-8
Best Generation	7	7
Final Generation	27	27
Objective Functions	Best Value	8
Design Variables (1)	Best Value	1
Design Variables (2)	Best Value	1
Constraints (1)	Best Value	0
Constraints (2)	Best Value	0
Constraints (3)	Best Value	0

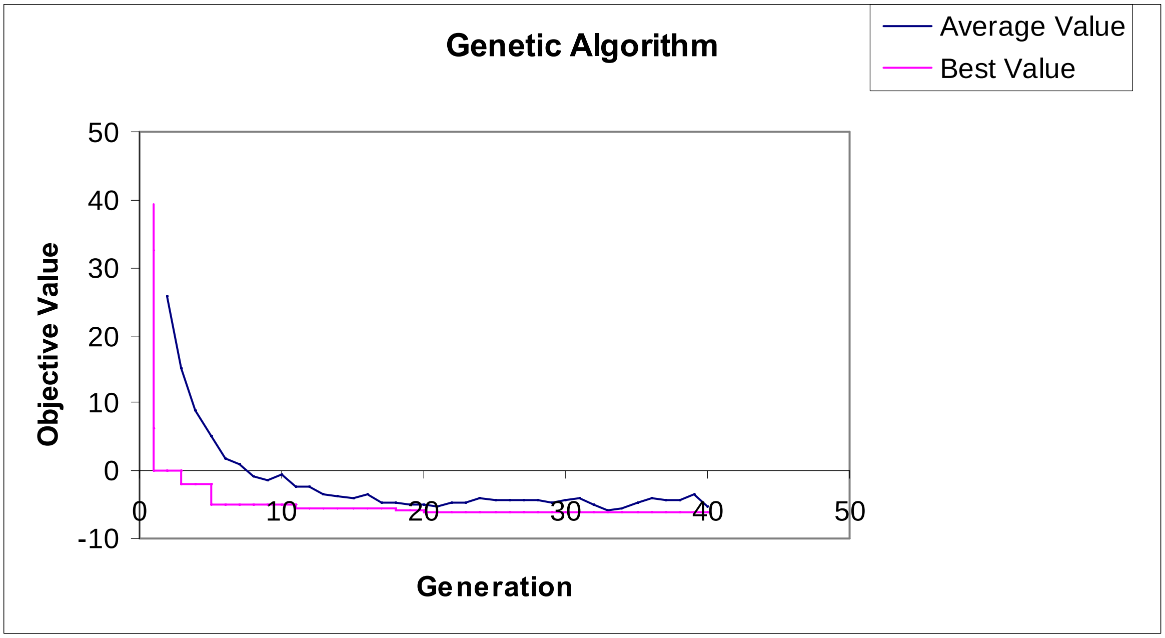
Figure 4.1: The objective function for instance of 30 trips



output for instance of 50 trips

OptWorks Optimizer Outputs		
Algorithm	Genetic Algorithm	Value
Best Net Objective Value		-6.16
Best Generation		20
Final Generation		40
Objective Functions	Best Value	6.16
Design Variables (1)	Best Value	1
Design Variables (2)	Best Value	1
Constraints (1)	Best Value	0
Constraints (2)	Best Value	0
Constraints (3)	Best Value	0

Figure 4.2: The objective function for instance of 50 trips



CHAPTER 5

CONCLUSIONS

In this paper, we provided the mathematical formulation and solution techniques that have been successfully used in most literatures of general university timetabling problems to generate a pre-planned timetable for the taxi industry in South Africa, focusing on the taxi ranks between Polokwane and Mankweng Township, in the Limpopo Province. New formulations of integer programming for taxi timetabling problems were presented. In our model a schedule for a taxi is composed of taxi blocks, where each block is constitutes a departure from a specified rank to another specified rank, and back to the original rank. Ideally, a taxi block comprises of two trips, where a trip starts from a specified rank (say A), to another specified rank (say B).

The formulation of our problem is based on information collected from the current status regarding the operations of taxis between the two depots (Polokwane and Mankweng Township). We have proposed a two-phase approach that consists of assigning a taxi (vehicle) block at each rank and incorporating all timetables from two different ranks to make one timetable.

In this study the proposed solution methods managed to produce a timetable with a non-conflicting set of taxis and no consecutive assignment of one taxi to the trips within a duration time (total time traveled on a return trip between two locations). The complexity of the various stages of the sub-problems under consideration was determined, based on assumptions made and related studies in the literature. The unpredicted events and passenger flow were taken into consideration. The current timetable becomes infeasible due to huge passenger flow that causes delay and the unpredicted events. The rescheduling process was considered to cope with the unpredicted events and delay caused by the passenger flow. The rescheduling model needs to adjust the current timetable in an effective that might have an effect on the recovery of ld need more detailed studies.

Some of the algorithms that proved to be successful in solving general timetabling problems use a two-stage approach where feasibility of the timetable was first obtained, and then optimality sought by different search methods. In particular, the GA was used in the research, mainly due to its flexibility and power to produce the best solution to timetabling problems. The algorithm starts with a population of a feasible search space. Two operators: mutations and crossovers were designed in such a way that they do not produce infeasible offspring. The nature of the problem under study also required some applications of Integer Programming. Since the complexity of problems similar to the one under study has already been proved to be NP-hard, it was also necessary to consider some heuristics such as SA.

While the focus of our research is on a single link, we provided an overview on various links for future research or other depots elsewhere in the country or beyond.

[1] Alverez-Valdes R., Crespo E. & Tamarit J.M. (2002). Design and Implementation of a course scheduling system using Tabu Search. *European Journal of Operations Research Vol.* ***137,*** *pp. 512-523.*

[2] Arntzen H., & Løkketangen A. A Local Search heuristic for a university timetabling problem.

www.idsaia.ch/files/ttcomp2002/arntzen.pdf.

(accessed on 23 November 2007)

[3] Batenberg K.J., & Palenstijn W.J. A New Examination Algorithm.

www.visielab.ua.ac.be/staff/batenburg/papers/bapa_bnaic_2003.pdf.

(accessed on 23 July 2007).

[4] Burke E.D., Bykov Y., Newall J. & Petrovic S. (2003). A time-predefined approach to course timetabling. *Yugoslav Journal of Operational Research. Vol.* ***13****, No. 2, pp. 139-151.*

[5] Burke E.D., Elliman D.G. & Weare F.R. (1995). A Hybrid Genetic Algorithm for Highly Constrained Timetabling Problems.

www.asap.cs.nott.ac.uk/publication/pdf/icga95.pdf.

(accessed on 14 March 2005).

[6] Burke E.K., Newall J.P., & Weare R.F. A Memetic Algorithm for University Exam timetabling problem.

www.cs.nott.ac.uk/~sxp/scheduling/univTTPatat.pdf.

(accessed on 10 May 2006)

[7] Caprara A., Monaci M., Toth P, Guida P.L. (2006). A Lagrangian heuristic algorithm for a real world train timetabling problem. *European Journal of Operational Research Vol.* ***148,*** *pp. 738-753.*

ristic Approach to Constraint Optimization in *Nat. Sci. Vol.* ***20****, pp. 64-67 Section B: short Communication.*

[9] Chang S-C., & Chung Y-C. (2005). From timetabling to train regulation-a new train operation model. *Information and Software Technology Vol. 47, pp. 575-585.*

[10] Chiarandini M., Birattari M., Socha K, & Doria O.R. (2006). An effective hybrid algorithm for university course timetabling. *Journal of Scheduling. Vol. 9, Issue 5, pp. 403-432.*

[11] Cheng E., Kleinberg R., Kruk S., Lindsey W., & Steffy D. (2004). A Strictly Combinatorial Approach to the University Exam Schedulinh Problem. www.isye.gatech.edu/~dsteffy/Files/TTCPaper.pdf. (accessed on 17 March 2008).

[12] Claessens M.T., Van Dijk N.M., & Zwaneveld P.J. (1998). Cost optimal allocation of rail passenger lines. *European Journal of Operational Research Vol.* ***110,*** *pp. 478-489.*

[13] Daskalaki S., & Birbas T. (2005). Efficient solution for university timetabling problem through integer through programming. *European Journal of Operational Research Vol.* ***160,*** *pp. 106-120.*

[14] Daskalaki S., Birbas T. & Housos E. (2004). An Integer Programming formulation for a case study in University Timetabling. *European Journal of Operational Research Vol.* ***153,*** *pp. 117-135.*

[15] Datta D. & Deb K. Solving Timetabling Problem of IIT Kanpur using Multiple-Objective Evolutionary algorithm.

www.iitk.ac.in/kangal/papers/k2006006.pdf.

(accessed on 10 August 2007).

[16] De Palma A. & Lindsey R. (2001). Optimal timetables for public transportation. *Transportation Research Part B* ***35,*** *pp. 789-813.*

G, Cunha J.F.E. Interactive Multi-objective he Bus Driver Scheduling problem. *Advanced OR and AI Methods in Transportation.*

www.iasi.cnr.it/ewgt/16conference/ID89pdf.

(accessed on 10 August 2007).

[18] De Werra D. (1985). An introduction to timetabling. *European Journal of Operational Research Vol.**19,** pp. 151-162.*

[19] De Werra D. (1997). The combinatorics of timetabling. European *Journal of Operations Research Vol.* ***96,*** *pp. 504-513.*

[20] Deris S., Omatu S.& Ohta H. (2000). Timetable Planning using the Constraint-based reasoning. *Computer & Operational Research Vol.* ***27,*** *pp. 819-840.*

[21] Desaulniers G., Lavigne J. & Soumis F. (1998). Multi-depot vehicle scheduling problems with time windows and waiting costs. *European Journal of Operational Research Vol.* ***111****, pp. 479-494.*

[22] Dias T.M., Ferber D.F., De Souza C.C. & Moura A.V. (2003). Constructing nurse schedules at large hospitals. *International Transportations In Operational Research. Vol.**10,** pp. 245-265.*

[23] Ebben M.J.R., Van der Heijden M.C., & Van Harten A. (2005).

Dynamic transport Scheduling under multiple resource constraints. *European Journal of Operational Research Vol.* ***167,*** *pp. 320-335.*

[24] Freling R, Wagelmans A.P.M, Paixao J.M.P. (2001). Models and Algorithms for Single ïDepot vehicle Scheduling. *Transportation Science Vol. 35, No.2, pp. 165-180.*

[25] Gaspero L.D, & Schaerf A. Tabu Search Techniques for Examinition Timetabling. www.diegm.uninud.it/satt/papers/DiSc00b.pdf

(accessed on 21June 2007).

& Suhl L. A variable fixing heuristic for the ted vehicle and crew scheduling problem. *Advanced Operations Research and AI methods in Transportation.* www.iasi.cnr.it/ewgt/16conference/ID120.pdf

(accessed on 20 April 2006)

[27] Glover F. (1988). Tabu Search. *CAAI Report 88-3.* University of Colorado.

[28] Gunadhi H., Anand V.J. & Yong Y.W. (1996). Automated Timetabling Using an Object-oriented Scheduler. *Expert Systems with applications . Vol.* ***10****, No. 2, pp. 243-256.*

[29] Gyori S., Petres Z., & Varkonyi-Koczy A.R. (2001). Genetic Algorithms in Timetabling. A New Approach. www.mft.hu/hallg/200107.pdf. (accessed on 26 June 2005).

[30] Huisman D, Wagelmans A.P.M. (2004). A solution approach for dynamic vehicle and crew scheduling. *European Journal of Operational Research Vol. 172*, *pp. 453-471.*

[31] Liam T.G. M., Boland N., Barry D.H. & Stuckey P.J. (2002). A Hybrid Algorithm for the Examination Timetabling problem. www.cs.mu.oz.au/~pjs/merlotPATA2002.pdf.

(accessed on 26 March 2006).

[32] Malachy C. & Sinead C. *(*2003). Scheduling and platforming trains at busy Complex stations. *European Journal of Operational Research Vol.* ***37****, pp. 195-224.*

[33] Mauri G.R., & Lorena L.A.N. (2004). Driver Scheduling Generating Using a Population training Algorithm. Proceeding of I Brazilian workshop on Evolutionary Computation, In SBRNò04 ï Brizilian Symposium in Neural networks. São Luis, Maranhão ï Brazil.

www.lac.inpe.br/~lorena/mauri/driver-pta.pdf.

ber 2007)

[34] Muller T. (2002). Some Novel Approaches to Lecture Timetabling. www.ktiml.ms.mff.cuni/cz~muller/t02-na.pdf

(accessed on 31 July 2007).

[35] Mushi A.R. (2004). Mathematical Programming Formulations for the Examinations Timetable Problem: The case of the University of Daressalaam. *African Journal of Science and Technology. Vol. 5, No. 2, pp. 34-40.*

[36] Mushi A.R. (2006). Tabu Serach Heuristic for University Course Timetabling Problem. *African Journal of Science and Technology. Vol. 7, No.1, pp. 34-40.*

[37] Nthangeni M.E. (1986). An algorithm for a university class timetable. *M.Sc. Dissertation, Unisa, 1986.*

[38] Ranson D., & Ahmadi S. (2006). An Extensible Modeling Framework for the Examination Timetabling Problem.

www.patat06.muni.cz/doc/patat06_110.pdf.

(accessed on 03 August 2007).

[39] Rodrigues M.M, De Souza C.C, Moura A.V (2006). Vehicle and Crew scheduling for Urban bus lines. *European Journal of Operational Research Vol.* ***170,*** *pp. 844-862.*

[40] Sangheon H. (2004). A Hybrid Meta-Heuristic Algorithm for Vehicle Scheduling Problem. *NUCB Journal of Economics and information Science. Vol.* ***48,*** *No.2, pp. 343-358.*

[41] Schaerf A. Local Search Techniques for Large High-School Timetabling Problems.

www.diegm.uniud.it/satt/papers/Scha99c.pdf.

(accessed on 04 August 2007)

bi T, Kuljis J. (2006. A proposed Multiagent eduling.

www.informs-sim.org/wsc06paper/197.pdf.

(accessed on 7 August 2007)

[43] Sheibani K. (2002). An Evolutionary Approach for the Examination Timetabling Problems.

Ingenier.kahosI.be/vakgroep/IT/pata2002/ExaminationTT/sheibeni/.

(accessed on 20 June 2005).

[44] Taha H.T. (1989). Operations Research An introduction Forth Edition. Macmillan Publishing Co., New York.

[45] Wren A. & Wren D.O. (1995). A genetic algorithm for public transport driver scheduling. *Computer & Operational Research. Vol.* ***22****, No. 1, pp. 101- 110.*

[45] Yanez J. & Ramirez J. (2003). The robust coloring problem. *European Journal of Operational Research Vol.* ***148,*** *Issue. 3, pp. 195-224.*

[46] Yu E. & Sung K.-S. (2002). A genetic algorithm for university weekly courses timetabling problem. *International Transactions in Operational Research Vol. 9, pp 703-717.*

Preliminary data from the Polokwane ï Mankweng Taxi Association.

1. Number of taxis in the association											
2. Number of taxis (seat capacity):											
2.1. 10 seater capacity		2.3. Other seater capacity (specify)									
2.2. 15 seater capacity		___________________									
3. Working hours	3.1. Starting time		3.2. Ending time								
4. Do you change drivers between these hours	No		Yes								
4.1. If Yes, How many hours are allocated to each driver per day?											
5. Estimate the number of passengers during rush hour that take taxi at Mankweng taxi rank (e.g. 06:00-08:00, 16:00-18:00)											
6. Number of taxis assigned for Mankweng taxi rank during rush hours											
7. Estimate the number of passengers during rush hours that take taxi at Polokwane taxi rank (e.g. 06:00-08:00, 16:00-20:00)											
8. Number of taxis assigned for Polokwane- Mankweng taxi rank during rush hours											
9. Estimate the number of passengers during rush hours that take taxi at intermediate stops											
10. Estimate the number of taxis assigned for passengers that are at stops during rush hours (floating taxis)											
11. Number of trips assigned for each taxi per day											
12. Traveling time between Mankweng and Polokwane											
13. Idling time at taxi rank for taxi to be filled up	Rush hours		Off peak								
15. How long should taxis be scheduled before/ after another	Rush hours		Off peak								
16. If taxis operate differently	Days	No of taxis									
	16.1. Monday to Friday										
	16.2. Saturday										
	16.3. Sunday										
17. Number of accidents that occurred in the past twelve years in which taxis were involved											
1994		1995		1996		1997		1998		1999	
2000		2001		2002		2003		2004		2005	

re found guilty during accidents in the past twelve

Year	Reasons (Specifying the number of accidents caused by the following reasons.)			
	Drunkenness	Loud Music	Overloading	High Speed driving
1994				
1995				
1996				
1997				
1998				
1999				
2000				
2001				
2002				
2003				
2004				
2005				

19. Number of members were registered in association each year

1994		1995		1996		1997		1998		1999	
2000		2001		2002		2003		2004		2005	

Information provided by:

Name :_________________

Signature :_________________

Date :_________________

Stamp