

DYNAMIC TIMETABLE GENERATOR USING
PARTICLE SWARM OPTIMIZATION (PSO)
METHOD

TEH YUNG CHUEN

UNIVERSITY MALAYSIA PAHANG

DYNAMIC TIMETABLE GENERATOR USING
PARTICLE SWARM OPTIMIZATION (PSO) METHOD

TEH YUNG CHUEN

CB10045

FACULTY OF COMPUTER SYSTEM & SOFTWARE ENGINEERING

UNIVERSITY MALAYSIA PAHANG

MADAM ROZLINA MOHAMED



UNIVERSITI MALAYSIA PAHANG

BORANG PENGESAHAN STATUS TESIS

JUDUL:

SESI PENGAJIAN:

SAYA(HURUF BESAR)

Mengaku membenarkan tesis/laporan PSM ini disimpan di Perpustakaan Universiti Malaysia Pahang dengan syarat-syarat kegunaan seperti berikut:

1. Tesis/Laporan adalah hakmilik Universiti Malaysia Pahang.
2. Perpustakaan Universiti Malaysia Pahang dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan tesis ini sebagai bahan pertukaran antara institut pengajian tinggi.
4. **Sila tandakan (√)

SULIT

(Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972) *

TERHAD

(Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi/badan di mana penyelidikan dijalankan) *

TIDAK TERHAD

Disahkan Oleh

.....

.....

Alamat tetap:

Penyelia:

Tarikh:

Tarikh:

*Sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali sebab dan tempoh tesis/laporan ini perlu dikelaskan sebagai SULIT atau TERHAD.

DECLARATION

I hereby declare that this thesis entitled “Dynamic Timetable Generator Using Particle Swarm Optimization (PSO) Method” is the result of my own research except as cited in the references. The report has not been accepted for any degree and is not concurrently submitted in candidature of any degree.

Signature :.....

Name : Teh Yung Chuen

Matric No : CB10045

Date :

SUPERVISOR'S DECLARATION

“I hereby declare that I have read this thesis and my opinion this thesis is sufficient in terms of scope and quality for the submission of PSM 2, Degree in Computer Science (Software Engineering)”

Signature :

Supervisor : Dr. Rozlina Binti Mohamed

Date :

ACKNOWLEDGEMENT

First and foremost, I would like to thank the one and only God for giving me patience and good health throughout the duration of this Degree research. Other than that, I would like to express my gratitude to my supervisor, Madam Rozlina Mohamed for her advices, guidance, and support for me to finish up my PSM 2 in this semester. I also like to thank my respectful academic advisor, Encik Mohd Hafiz Bin Mohd, who gives me advice on my academics whenever I faced difficulties in my academics.

Moreover, I would like to thank University Malaysia Pahang for giving me the chance to further my studies as a degree student and learn more knowledge which is helpful for my future. I deeply thank to all the Fakulti Sistem Komputer Kejuruteraan Perisian's staff for helps me during my study time.

Lastly, I would also like to thank to my family and fellow friends who gave me support in everything. I will never forget those who help me out in my studies. Thank you all for spend so much time to help me. Thanks.

ABSTRACT

This paper addresses the usage of Particle Swarm Optimization (PSO) in generating a timetable which the selection of driver and vehicle are based on the concept of PSO. The objectives are simultaneously considered as follow: 1) minimizing the cycle time, 2) regenerate the timetable. Searching for an optimal solution in such of large sized population will be time consuming and thus by presenting the PSO method is able to select the appropriate driver and vehicle with a shorter period. The timetable that is generated will be more appropriate as regenerating function can handle emergency such as breakdown of vehicle. Besides, during the generating of timetable, it also considers constraints which make the task more challenging. The chosen particle during implementing the PSO method should be chosen with fitness nearest to fifty in this system. Thus, the timetable for transport schedule system can be arranged without clashing of driver or vehicle.

ABSTRAK

Dalam kertas ini mengatakan bahawa kegunaan konsep “Particle Swarm Optimization (PSO)” dalam memilih seorang pemandu dan kenderaan untuk setiap perjalanan. Objektif untuk tajuk ini adalah dianggap seperti berikut: 1) mengurangkan tempoh masa, 2) menjana semula jadual. Dalam populasi yang besar, cara penyelesaian masalah yang optimum akan menggunakan masa yang lama. Jadi, cara PSO telah dikemukakan untuk memilih pemandu dan kenderaan yang sesuai dalam masa yang lebih pendek. Selain itu, jadual yang telah dihasilkan boleh digenerasi semula jika ada sebarang kecemasan berlaku seperti kerosakan kenderaan. Tambahan pula, semasa penjaduan jadual tersebut, kekangan juga akan diambil kira yang menyebabkan tugas penghasilan jadual ini lebih mencabar. Semasa kaedah PSO ini dijalankan, zarah yang mempunyai nilai yang berdekatan dengan lima puluh akan dipilih. Justeru, jadual untuk sistem jadual pengangkutan boleh diatur tanpa bertempur pemandu atau kenderaan.

TABLE OF CONTENTS

	Page
DECLARATION	ii
ACKNOWLEDGMENTS	iv
ABSTRACT	v
ABSTRAK	vi
CONTENTS	vii
LIST OF TABLES	x
LIST OF FIGURES	xi
LIST OF ABBREVIATIONS	xii
LIST OF APPENDICES	xiii

CHAPTER I	INTRODUCTION	1
1.1	Introduction	1
1.2	Problem Statement	2
1.3	Scope	2
1.4	Objectives	3
1.5	Thesis Organization	3
CHAPTER II	LITERATURE REVIEW	4
2.1	Introduction	4
2.2	Background of Study	5
2.3	Artificial Intelligence Techniques	5
2.4	Particle Swarm Optimization (PSO)	5
	2.4.1 The Flow of Particle Swarm Optimization	6
	2.4.2 Particle Swarm Optimization Process	6
	2.4.3 Constriction Factor Version of Particle Swarm Optimization	7
2.5	Heuristic Method	7
2.6	Constraints	8
2.7	Comparison of Heuristic Method and Particle Swarm Optimization (PSO) Method	9
2.8	Conclusion	10

CHAPTER III	METHODOLOGY	11
3.1	Introduction	11
3.2	Feasibility Study	11
3.3	Methodology	12
3.3.1	Modified Rapid Application Development	12
3.3.2	Planning Phase	13
3.3.3	Iterative Development Phase	13
3.3.4	Deployment Phase	13
3.4	Development of Particle Swarm Optimization (PSO)	14
CHAPTER IV	DESIGN AND IMPLEMENTATION	15
4.1	Introduction	15
4.2	System Interface	15
4.3	Coding Structure	19
4.3.1	Database Architecture	19
4.3.1.1	admin Table	19
4.3.1.2	bkp_driver Table	20
4.3.1.3	bkp_vehicle Table	20
4.3.1.4	btm_driver Table	20
4.3.1.5	btm_vehicle Table	21
4.3.1.6	msh_driver Table	21
4.3.1.7	msh_vehicle Table	22
4.3.1.8	trip_bkp	23
4.3.1.9	trip_btm	23
4.3.1.10	trip_msh	23
4.3.2	Verification	24
4.3.3	Timetable Coding Structure	25
4.3.4	Particle Swarm Optimization (PSO) Algorithms	26
4.3.4.1	Hard Constraints	27
4.3.4.2	Implementation of Particle Swarm Optimization (PSO) Algorithms	27
4.4	Conclusion	29

CHAPTER V	RESULT AND DISCUSSION	30
5.1	Introduction	30
5.2	Result and Discussion	30
5.3	Advantage	33
5.4	Limitation	33
5.5	Contribution	33
5.6	System testing	34
5.7	Conclusion	34
CHAPTER VI	CONCLUSION	35
6.1	Introduction	35
6.2	Result Analysis	35
6.3	Future Work	36
6.4	Conclusion	36

LIST OF TABLES

TABLE NO	TABLE TITLE	PAGE
2.1	Comparison of Particle Swarm Optimization (PSO) Method	9
4.1	Data Dictionary of admin Table	19
4.2	Data Dictionary of bkp_driver Table	20
4.3	Data Dictionary of bkp_vehicle Table	20
4.4	Data Dictionary of btm_driver Table	20
4.5	Data Dictionary of btm_vehicle Table	21
4.6	Data Dictionary of msp_driver Table	21
4.7	Data Dictionary of msp_vehicle Table	22
4.8	Data Dictionary of trip_bkp Table	22
4.9	Data Dictionary of trip_btm Table	23
4.10	Data Dictionary of trip_msp Table	23
5.1	Tabulated list of random driver selected.	31

LIST OF FIGURES

FIGURE NO	FIGURES TITLE	PAGE
2.1	Pseudo Code of Heuristic Method	8
3.1	Rapid Application Development Model (RAD)	12
4.1	User Login Interface	15
4.2	Error Message for Invalid Login	16
4.3	Timetable in Bahagian Khidmat Pengurusan	16
4.4	Interface for adding a trip	17
4.5	Trip added in Bahagian Khidmat Pengurusan	17
4.6	Detail of driver in Bahagian Khidmat Pengurusan	18
4.7	Detail of vehicle in Bahagian Khidmat Pengurusan	18
4.8	Code that connect with database cb10045 using SQL command	19
4.9	Sample verification code to verify the valid IC number input	25
4.10	Sample code for 1.00 timeslot that loop from Monday to Sunday	26
4.11	Sample code for checking availability of driver	27
4.12	Sample code of choose driver randomly	27
4.13	Sample code of calculating fitness for each driver	28
4.14	Sample code of calculating velocity for each driver	29
5.1	Arrangement of trip which does not clash in a particular timeslot	30
5.2	List of driver that has been chosen randomly	31
5.3	Detail of trip before the trip is regenerated	32
5.4	Detail of trip after the trip is regenerated	32

LIST OF ABBREVIATIONS

- UMP – University Malaysia Pahang
PSO – Particle Swarm Optimization
RAD – Rapid Application Development

LIST OF APPENDICES

APPENDIX	TITLE	PAGE
A	Project Gantt Chart	38
B	Hierarchy Chart for Interface Design	40
C	System Testing	41

Chapter I

Introduction

1.1 Introduction

Timetabling problems have been discovered in these recent years and obviously there is an increased level of research activity in this area. There are variety types of timetabling problems arise such as educational timetabling, nurse timetabling and transportation timetabling. Timetabling is an arrangement of time for some events so that it fulfills the tasks given in the event. Timetabling is important to make sure the flow of the timetable can be smooth and success and at the meantime it does not occur any time conflict between each other.

In this transportation timetabling, the condition of drivers and vehicles need to consider so that the timetable generated can fulfill the needs of the user. In this case, an approach which is suitable to solve this situation is preferred to handle any sudden changes. In addition, the number of passengers will be considered for every trip from one station to other station so that it can fit all the passengers listed for every trip.

Particle Swarm Optimisation (PSO) is a self-adaptive global search optimization techniques introduced by J. Kennedy and R. Eberhart[1]. PSO is an artificial intelligence technique which is used to find an approximate solution in a numeric maximization and minimization problems. The model of PSO is most likely as the nature of bird flocking and fish schooling. The concept of PSO is implemented using the stochastic nature of the particle and it's converging to global with a reasonable solution on the particular problem. PSO is well-known and common in the artificial intelligence field due to its effectiveness in wide range of application, simplicity and lower cost.

1.2 Problem Statement

There is a need to consider the availability of drivers and vehicles of a bus station in order to generate a suitable timetable. For an example, if the driver is available while the vehicles are not available of a particular bus station, then the timetable need to be rearranged so that it can fulfill the task given. Therefore, the changes in drivers and vehicles availability need to be considered so that a replacement can be made immediately in order to run the timetable successfully.

With the method PSO used, it will generate a timetable through the selection of driver in random with certain condition and match with the vehicles available and come out with a satisfied result. In order to have the result which is satisfy by the user, some constraints need to be fulfilled such as the availability of drivers and the condition of vehicles to prevent the delay of the departure time. In order to present the result in satisfactory, the searching for the driver and vehicle is not only in the particular station but also other station from other area which means distributed driver and vehicles in term of PSO. However, heuristic method is not applicable with a distributed driver and vehicles.

1.3 Scope

The timetable is generated based on the availability of driver and vehicles in a particular station. The process is generated through a method call Particle Swarm Optimization (PSO) which will search for the driver and vehicles that available for the timetable. In case, if a driver is sick suddenly for that day, it will make replacement automatically from the list of other driver so that the timetable can run as usual. Besides, the number of passengers is also considered in order to choose the bus which can fit all the passengers. The appropriate selection of vehicles according with the number of passengers will be made.

1.4 Objectives

1. To generate a timetable in a system by using the PSO method.
2. To enables the user make replacement of driver or vehicle using the PSO method if necessary.

1.5 Thesis Organization

Thesis organization explains about the content that is included in every chapter so that the user can get the rough idea for each chapter. In Chapter 1 Introduction, explanation about the project is written to the users so that users can roughly get the ideas of the proposed project.

In Chapter 2 Literature Review, references are found for the project in order to makes sure that the project is done with prove.

In Chapter 3 Methodology, method used for the project will be discussed in order to get the framework of the project.

In Chapter 4 Design, this phase of the thesis is to develop out the framework or system of the project to the user.

In Chapter 5 Implementation, at this phase the workout of the project is being developed and the processes involved is recorded.

In Chapter 6 Results and Discussion will discuss about the results that being implemented to make sure that the outcome is fulfill the user need.

In Chapter 7 Conclusion is to make the summary about the project and the research that have been done.

Chapter II

Literature Review

2.1 Introduction

The problem for this paper has its origin in bus timetabling system for a particular bus station. It requires the selection of a driver and vehicles so that it can produce a timetable which satisfies the requirements Besides, the number of passengers for a slot of time is also considered so that the type of vehicles is chosen appropriately for each slot of time. By assigning the selected driver and vehicles into a particular time slot need to be considered thoroughly and along with the frequency of the service so that the driver get enough rest before the scheduled timetable start.

Particle Swarm Optimisation (PSO) is a self-adaptive global search optimisation technique introduced by Kennedy and Eberhart which has the algorithm that similar to other population-based algorithms like Genetic algorithms but there is random combination of individuals of the population [2].

2.2 Background of Study

There are a lot of research have been carried out about the methods of generating a timetable using different kind of algorithms. A good timetabling of an event can always make sure the flow of the event to run smoothly and there is always a challenging issue for the developer to create a timetable using the appropriate artificial intelligence technique. In this case study, there are a lot of field to be studied such as the educational timetabling [3], sport timetabling [4] and transportation timetabling [5].

In transportation timetabling, the factors that need to be considered are drivers and vehicles to make an optimum timetable. Due to the constraint of the working hour of a driver along with other constraints, scheduling has its difficulty to be created with the constraint included.

Driver constraint that insisted the most is the working hour per day. The driver should not work continuously without any rest as they need to drive from one station to another station safely. The constraint of vehicles is the condition of the vehicle so that it will not happen any accident along the trip. Besides, the size of a vehicle should also be considered so that the number of passengers is able to fit into the vehicle.

2.3 Artificial Intelligence Technique

Artificial Intelligence (AI) technique is a language which can only be read by the computer and the code that deploy by the programmer into the computer will be executed. The program will take the input from the user and display the result in the form of number after the constraints have been included. The different type of AI techniques will display different type of result as the decision made is altered with the constraints included and based on the user demand.

2.4 Particle Swarm Optimization

The particle swarm optimizer (PSO) algorithm is first present by Dr. Kennedy and Dr. Eberhart, and is a random evolution method based on intelligent search of the group birds. It has quick convergence speed and optimal searching ability for solving large-scale optimization problems [6]. Particle Swarm Optimization (PSO) is a computational method which simulates a problem with particles and moving these particles in a given search space based on a mathematical formulae over the particle's position and velocity.

2.4.1 The Flow of PSO

In Particle Swarm Optimization (PSO), the problem is solved by defining a number of particles in a search space for the processing of the objective function at its current position. The movement of the particles will be determined by combining it with the best fitness location randomly.

The dimension of PSO can be classified into three parts which are current position x_i , previous best position p_i , and the velocity v_{id} . For each individual in particle swarm will possess the three D-dimension of vectors, the D refers to the dimension of the search space.

As the iteration starts, it will evaluate the current position x_{id} as the problem solution. The current position x_i will contain the coordinate of a particle in the search space. Once the iteration ends, it will determine whether the current position of the particle is still the best position. If it is, the position will be saved for the next vector p_{id} .

After the position has been saved, in the next iteration, it will be named as previous best position $pbest_{id}$ and taken for the next iterations in order to get a better result. As the iteration keeps going, it will keep updating the value of the p_{id} and $pbest_{id}$.

At this moment, the vector v_i will be adjusted and add to the current position x_{id} . Hence, a new point will be selected. Once the local iteration ends, if the vector p_{id} calculated is the best among the local then it will be compared among the global. If the vector is the best among the global then it will automatically be chosen as the best among global p_{gd} .

2.4.2 Particle Swarm Optimization Process

1. Initialize population in search space.
2. Evaluate fitness of individual particles.
3. Modify velocities based on previous best and global (or neighborhood) best.
4. Terminate on some condition.
5. Go to step 2.

$$v_{id} = w_{id}v_{id} + c_1rand()(p_{id} - x_{id}) + c_2Rand()(p_{gd} - x_{id})$$

$$x_{id} = x_{id} + v_{id}$$

where d is the dimension,

c_1 and c_2 are positive constants,

rand and Rand are random function,

w is the inertia weight,

v_i is the velocity,

x_i is the current position,

p_i is the previous local position,

p_g is the previous global position.

2.4.3 Constriction Factor Version of PSO

$$v_{id} = K * [w_{id}v_{id} + c_1 \text{rand}() (p_{id} - x_{id}) + c_2 \text{Rand}() (p_{gd} - x_{id})]$$

$$K = 2 / |2 - \varphi + \sqrt{\varphi^2 - 4\varphi}|$$

2.5 Heuristic Method

In the field of computer science, heuristic method is an artificial intelligence which is designed to solve a problem by finding the approximate solution when the existed method fails to find the exact solution. The purpose of heuristic method is to produce a solution for a problem which approximately same with the exact solution. The result from the heuristic method may not be the best but due to its time consumed to generate the solution is short and thus the result is still acceptable as the result approximate to the exact solution.

In heuristic method, there are two types of conditions which are: a set of variables that are not being attached and the half-way solution. For each iterations of the algorithms, it will try to convert the half-way solution into a complete solution which can solve the given problem. Once the algorithms start with the existed solution, it will loop until it reaches the maximum iterations or it has assigned all the variables.

```

procedure solve(unassigned, solution, max_iter)
  // unassigned is a list of un-assigned variables
  // solution is a partial solution (empty at the beginning)
  //   e.g. a list (variable, assigned value)
  iterations=0;
  while unassigned non empty & iterations<max_iter
    & non user interruption do
    iterations ++;
    variable = selectVariable(unassigned, solution);
    unassigned -= variable;
    value = selectValue(variable, solution);
    unassigned += assign(solution, variable, value)
    // assign the variable and return violated variables
  end while;
  return solution;
end solve

```

Figure 2.1 Pseudo Code of Heuristic Method

The algorithms choose the value by passing through the variables. By going through each domain via the function stated in the algorithms, the value of the variable will be selected. The loop will continue until it finds the best fitness value.

2.6 Constraint

A hard constraint is the rule that need to be satisfied in a system. In the development of a system, the hard constraint needs to be stated clearly so that it will produce an optimum result. Time clashing is the hard constraint that needs to be fulfilled in this system. Both driver and vehicle are supposed not to be assigned into a same timeslot. In order to prevent time clashing, both driver and vehicle that have been assigned for work will not be selected in the process.

A soft constraint is the rule that should be considered in a system. The soft constraint in this project is the working hours of the driver. Although a driver can be assigned for work in a continuous timeslot, it is considerable that offer a resting timeslot for a driver after a particular timeslot. The resting hour of the driver will be considered depending on how many hours the driver has been worked for.

2.7 Comparison of Heuristic Method and Particle Swarm Optimization

Table 2.1 Comparison of Particle Swarm Optimization (PSO) Method and Heuristic Method

	Particle Swarm Optimization (PSO)	Heuristic Method
Simulation	Birds Flocking in a population	Mathematical Algorithms
Process	<ol style="list-style-type: none"> 1. Generating particles position and velocity. 2. Update the velocity and the position. 3. Retrieve the best global position. 	By evaluating the function, the search for the best solution is made in the algorithms. The evaluation function will update the existed solution to a better solution through the iteration in the algorithms.

The process of Heuristic method and Particle Swarm Optimization (PSO) is almost the same as both methods are also evaluating the function in order to search for the best solution of the problem. However, in PSO, there are simulating the problem as the particles in a search space and given their current best position in the search space. After that, it will generate the solution by adding velocity to the particle with current best position and a new position for the particles will be generated. The fitness value for the particles will be evaluated along with the velocity until it meets a condition to terminate the iteration and then the local best position will be generated. Next, the local best position will then be brought forward to be compared among the global to retrieve for the best global position which means the best solution among the population.

For heuristic method, there are search spaces as well but the process will go through all the possible solution to generate for a better solution. By applying the algorithms into the iterations, the function set will update the solution into a better solution and it will take time with the evaluating process.

2.8 Conclusion

As a result, in transport scheduling system, timetable generation should be efficient in term of time and usability. By using Particle Swarm Optimization (PSO) method, the timetable can be generated by choosing the particle (driver or vehicle) randomly so that it does not search one by one for the selection of driver or vehicle. Compared to Heuristic Method, the timetable that is being generated will need to search for every driver before a driver is selected for a trip which is not efficient. Thus, Particle Swarm Optimization method is chosen for generating the timetable of this Transport Scheduling System.

Chapter III

Methodology

3.1 Introduction

This chapter discuss about the methodology that is used for this system. Methodology in software development is used to control the process of a development. The type of methodology used is depend on the system as the correct used of methodology will efficiently produce the application.

3.2 Feasibility Study

Scheduling is an issue that is always been discussed in the computer field as arranging a timetable will be time consuming. It needs to consider a lot in order to create a timetable that can be used by the user without any clashing of time. A lot of timetabling studies have been carried out by using different kind of artificial intelligence techniques such as Graph Heuristic Method, Genetic Algorithms and so on. By applying the techniques, the system will help the user to create a timetable automatically after fulfil all the constraints that have been defined in the system.

3.3 Methodology

Methodology is the method used for a specific system. By having the flow of the methodology, the system is being developed from phase to phase and this will make sure that the system is being developed correctly before deployment. Methodology can be used as a guideline in a system so that the system can be developed with procedure. In this project, I would like to implement Rapid Application Development (RAD) method. With RAD method, the system will be developed efficiently as this method provide a more realistic and practical method during development.

3.3.1 Modified Rapid Application Development (RAD)

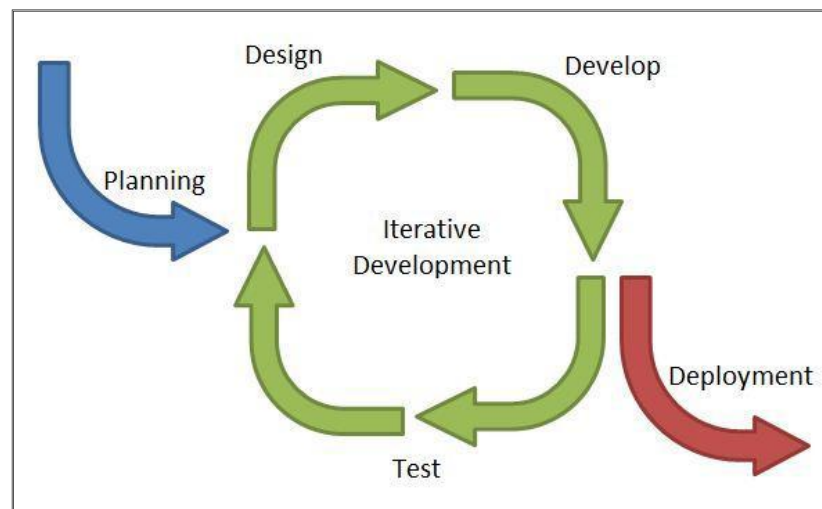


Figure 3.1 Rapid Application Development Model (Modified)

3.3.2 Planning Phase

This phase representing the starting phase of the system which requires understanding and getting the requirement of the system. During this phase, research has been done in order to be implemented into the system. Variety of artificial intelligence techniques and the requirements of the system have been studied for implementing into the system.

Besides, studies of timetabling for transportation which consists of suitable timeslot and working hour have been done in this phase. After that, selecting the suitable artificial intelligence techniques will be made after understanding the concept of conducting a timetable.

3.3.3 Iterative development Phase

The main purpose of this phase is to handle any changes of the requirements on the system. This phase is created as a form of cycle which consists of design, develop and test. At the beginning of the development, a prototype is design according to the user requirements and the flow of the system will be noted so that the development of the system can satisfy the user. After the sketch of the design is created, development phase will start in order to create a system which will be used by the user. Next will be the testing phase which will test for the usability and functionality of the system after the system has been developed. At this phase, the user can check the system whether the system fulfil their requirements.

3.3.4 Deployment Phase

This phase will only be done after the system is completely developed and full testing has been carried out on the system and this will make sure that the system that is created fulfil user requirements and does not contain major errors. This means that the system can be deployed at the user site.

3.4 Development of Particle Swarm Optimization

Let S be the number of particles in the swarm, each having a position \mathbf{x}_i in the search-space and a velocity \mathbf{v}_i . Let \mathbf{p}_i be the best known position of particle i and let \mathbf{g} be the best known position of the entire swarm. A basic PSO algorithm is then:

- For each particle $i = 1, \dots, S$ do:
 - Initialize the particle's position with uniformly distributed random vector: $\mathbf{x}_i \sim U(\mathbf{b}_{lo}, \mathbf{b}_{up})$, where \mathbf{b}_{lo} and \mathbf{b}_{up} are the lower and upper boundaries of the search-space.
 - Initialize the particle's best known position to its initial position: $\mathbf{p}_i \leftarrow \mathbf{x}_i$
 - If $(f(\mathbf{p}_i) < f(\mathbf{g}))$ update the swarm's best known position: $\mathbf{g} \leftarrow \mathbf{p}_i$
 - Initialize the particle's velocity: $\mathbf{v}_i \sim U(-|\mathbf{b}_{up}-\mathbf{b}_{lo}|, |\mathbf{b}_{up}-\mathbf{b}_{lo}|)$
- Until a termination criterion is met (e.g. number of iterations performed, or a solution with adequate objective function value is found), repeat:
 - For each particle $i = 1, \dots, S$ do:
 - For each dimension $d = 1, \dots, n$ do:
 - Pick random numbers: $r_p, r_g \sim U(0,1)$
 - Update the particle's velocity: $\mathbf{v}_{i,d} \leftarrow \omega \mathbf{v}_{i,d} + \phi_p r_p (\mathbf{p}_{i,d} - \mathbf{x}_{i,d}) + \phi_g r_g (\mathbf{g}_d - \mathbf{x}_{i,d})$
 - Update the particle's position: $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$
 - If $(f(\mathbf{x}_i) < f(\mathbf{p}_i))$ do:
 - Update the particle's best known position: $\mathbf{p}_i \leftarrow \mathbf{x}_i$
 - If $(f(\mathbf{p}_i) < f(\mathbf{g}))$ update the swarm's best known position: $\mathbf{g} \leftarrow \mathbf{p}_i$
 - Now \mathbf{g} holds the best found solution.

Assumptions:

search space is the table of driver in a database.

particle is simulating with the driver with its own position.

Chapter IV

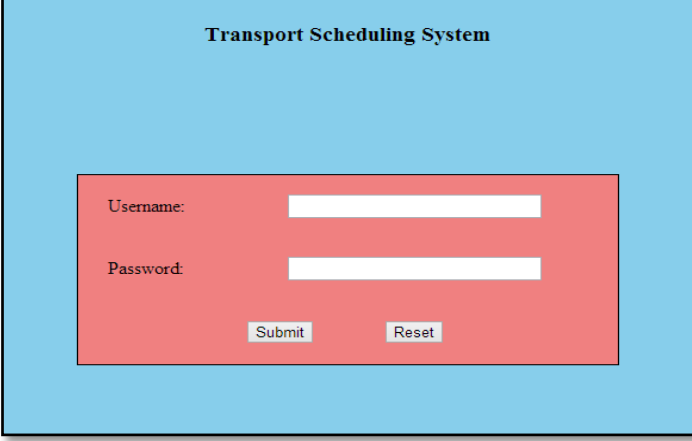
Design and Implementation

4.1 Introduction

This chapter will describe on how the system interface is designed and implemented. The system interface will be designed and the flow of the system is developed. Based on the Rapid Application Development methodology, the Development phase is conducted along with the implementation of the system. Once a part of the development is conducted, it will be implemented into the system. In this chapter, the implementation of the system interface design, database and source code will be reviewed.

4.2 System Interface

The Figure 4.1 below shows the login interface in order to verify the user identity. The user must login with their correct username and password before they use the function of the system.



The image shows a user login interface for a system titled "Transport Scheduling System". The interface is contained within a light blue rectangular frame. At the top center of this frame, the text "Transport Scheduling System" is displayed. Below this, there is a red rectangular area containing the login form. The form includes two input fields: "Username:" followed by a white text box, and "Password:" followed by a white text box. Below these fields are two buttons: "Submit" and "Reset", both with a light gray background and black text.

Figure 4.1 User Login Interface

If the user does not enter the valid username and password, it will pop out an error message box as shown in Figure 4.2 below indicate that the user has not signed up yet.

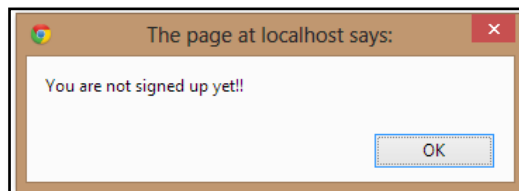


Figure 4.2 Error message for invalid login

The system will allow the user to view the trip added in the table format as shown in Figure 4.3 below. This will ease the user to check for the time of trip and day of the trip. In the table shown, user is able to view for the driver and vehicle assigned for the trip and the location of the trip.

Bahagian Khidmat Pengurusan							
Vehicle Driver Trip Timetable Department Logout							
Hours/Days	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
1:00							
2:00		Kuala Lumpur driver_1 Bus - CBB 3433		Selangor driver_2 Bus - CBB 3433			
3:00							

Figure 4.3 Timetable in Bahagian Khidmat Pengurusan.

In this Transport Scheduling System, user can add trip for the timetable. Interface below shows that user needs to fill in the detail of the trip before the trip can be added into the timetable. Once the user confirm with the detail of the trip, the system will execute the Particle Swarm Optimization (PSO) algorithms which helps the user to assign a driver and a vehicle for the trip.

Figure 4.4 Interface for adding a trip

After a trip has been added for a certain department, user can view the trip from the tab as shown in the Figure 4.5 below. The interface below is the trips that have been added for department of Bahagian Khidmat Pengurusan. Besides, user can also delete the trip added if necessary by clicking the “Cancel” button. For the “Regenerate” button, user is able to regenerate the timetable in order to assign a new driver and vehicle for the trip if there is any emergency.

Bahagian Khidmat Pengurusan									
Vehicle Driver Trip Timetable Department Logout									
Trip	Location	Driver	Vehicle	Type of Vehicle	Day	Time	Participant		
IBM Talk	Kuala Lumpur	Dern	CBC 3033	Bus	1	1	40	Cancel	Regenerate
Japanese Show	Kuala Lumpur	Aern	CBB 1236	Bus	1	1	40	Cancel	Regenerate
Japanese Show	Kuala Lumpur	Ern	CBB 2213	Bus	1	1	40	Cancel	Regenerate
Japanese Show	Malacca	Fern	CGV 4332	Bus	1	1	40	Cancel	Regenerate

Figure 4.5 Trip added in Bahagian Khidmat Pengurusan.

In the system, user can view or update the detail of the driver in a table. Figure 4.6 shows that the detail of driver in the department of Bahagian Khidmat Pengurusan.

Bahagian Khidmat Pengurusan							
Vehicle	Driver	Trip	Timetable	Department	Logout		
	Driver Name	IC Number	Gender	Age	Hand Phone Number	Status	
1	Aern	999999001111	Male	21	3333333333	No	<input type="button" value="Update"/>
2	Bern	000000110000	Male	26	2222000010	No	<input type="button" value="Update"/>
3	Cern	888888990000	Male	25	0000110000	No	<input type="button" value="Update"/>
4	Dem	111111002222	Male	25	1111111111	No	<input type="button" value="Update"/>

Figure 4.6 Detail of driver in Bahagian Khidmat Pengurusan

In each department, user is able to view for the detail of vehicle or update the existing vehicle in the particular department. Figure 4.7 shows the vehicle in department of Bahagian Khidmat Pengurusan. User can update the vehicle detail by clicking the “Update” button.

Bahagian Khidmat Pengurusan							
Vehicle	Driver	Trip	Timetable	Department	Logout		
	Department	Vehicle Plate Number	Status				
1	Bahagian Khidmat Pengurusan	CBB 1236	No				<input type="button" value="Update"/>
2	Bahagian Khidmat Pengurusan	CBB 5488	No				<input type="button" value="Update"/>
3	Bahagian Khidmat Pengurusan	CBC 3033	No				<input type="button" value="Update"/>
4	Bahagian Khidmat Pengurusan	CVD 2211	No				<input type="button" value="Update"/>

Figure 4.7 Detail of vehicle in Bahagian Khidmat Pengurusan

4.3 Coding Structure

In this section, the codes and function of the system will be reviewed.

4.3.1 Database Architecture

In Transport Scheduling System, database connection is needed in order to store the detail of the trip, driver and vehicle in a certain department. Besides, database is also used to save the fitness value of the selected driver or vehicle during Particle Swarm Optimization techniques.

```
<?php
$conn = mysql_connect("localhost","root","");
if(!$conn)
{
    die("Could not connect to database");
}
mysql_select_db("cb10045",$conn) or die("Could not open products database");
?>
```

Figure 4.8 Code that connect with database cb10045 using SQL command.

4.3.1.1 admin Table

Table 4.1 shows the table of an admin in the database that store the data of the admin. This will validate the admin before it can access into the system.

Table 4.1 Data Dictionary of admin Table.

Field Name	Type	Null
id	int(10)	No
username	varchar(20)	No
password	int(20)	No

4.3.1.2 bkp_driver Table

Table 4.2 shows bkp_driver Table that will contain information of a driver in Bahagian Khidmat Pengurusan.

Table 4.2 Data Dictionary of bkp_driver Table.

Field Name	Type	Null
d_id	int(11)	No
department	varchar(50)	No
d_name	varchar(30)	No
d_ic	varchar(15)	No
d_gender	varchar(10)	No
d_age	int(11)	No
d_hp	varchar(15)	No
status	varchar(5)	No
start_time	float	No
end_time	float	No
pfBest	float	No
position	float	No
velocity	float	No

4.3.1.3 bkp_vehicle Table

Table 4.3 shows bkp_vehicle Table that will have the information of a vehicle in Bahagian Khidmat Pengurusan.

Table 4.3 Data Dictionary of bkp_vehicle Table

Field Name	Type	Null
id	int(11)	No
department	varchar(50)	No
v_plate	varchar(10)	No
status	varchar(5)	No
start_time	float	No
end_time	float	No
pfBest	float	No
position	float	No
velocity	float	No

4.3.1.4 btm_driver Table

Table 4.4 shows btm_driver Table that has the information of driver which is from department of Bahagian Teknologi Maklumat.

Table 4.4 Data Dictionary of btm_driver Table.

Field Name	Type	Null
d_id	int(11)	No
department	varchar(50)	No

d_name	varchar(30)	No
d_ic	varchar(15)	No
d_gender	varchar(10)	No
d_age	int(11)	No
d_hp	varchar(15)	No
status	varchar(5)	No
start_time	float	No
end_time	float	No
pfBest	float	No
position	float	No
velocity	float	No

4.3.1.5 btm_vehicle Table

Table 4.5 shows btm_vehicle Table that will contain the information of each vehicle in Bahagian Teknologi Maklumat.

Table 4.5 Data Dictionary of btm_vehicle Table

Field Name	Type	Null
id	int(11)	No
department	varchar(50)	No
v_plate	varchar(10)	No
status	varchar(5)	No
start_time	float	No
end_time	float	No
pfBest	float	No
position	float	No
velocity	float	No

4.3.1.6 msp_driver Table

Table 4.6 shows msp_driver Table that will be used to save the information of driver in Majlis Sukan Pahang.

Table 4.6 Data Dictionary of msp_driver Table

Field Name	Type	Null
d_id	int(11)	No
department	varchar(50)	No
d_name	varchar(30)	No
d_ic	varchar(15)	No
d_gender	varchar(10)	No
d_age	int(11)	No
d_hp	varchar(15)	No
status	varchar(5)	No

start_time	float	No
end_time	float	No
pfBest	float	No
position	float	No
velocity	float	No

4.3.1.7 msp_vehicle Table

Table 4.7 shows msp_vehicle Table that will save the information of vehicle in Majlis Sukan Pahang.

Table 4.7 Data Dictionary of msp_vehicle Table.

Field Name	Type	Null
id	int(11)	No
department	varchar(50)	No
v_plate	varchar(10)	No
status	varchar(5)	No
start_time	float	No
end_time	float	No
pfBest	float	No
position	float	No
velocity	float	No

4.3.1.8 trip_bkp Table

Table 4.8 shows trip_bkp Table that will save the information of a trip in Bahagian Khidmat Pengurusan.

Table 4.8 Data Dictionary of trip_bkp Table.

Field Name	Type	Null
trip_name	Varchar(100)	No
trip_location	varchar(50)	No
trip_driver	varchar(50)	No
trip_vehicle	varchar(10)	No
trip_vtype	varchar(5)	No
trip_day	float	No
trip_time	float	No
end_day	float	No
end_time	float	No
trip_participant	int(11)	No

4.3.1.9 trip_btm Table

Table 4.9 shows trip_btm Table that will be used to save the information every trip in Bahagian Teknologi Maklumat.

Table 4.9 Data Dictionary trip_btm Table.

Field Name	Type	Null
trip_name	Varchar(100)	No
trip_location	vvarchar(50)	No
trip_driver	vvarchar(50)	No
trip_vehicle	vvarchar(10)	No
trip_vtype	vvarchar(5)	No
trip_day	float	No
trip_time	float	No
end_day	float	No
end_time	float	No
trip_participant	int(11)	No

4.3.1.10 trip_msp Table

Table 4.10 shows trip_msp Table that will save the information of each trip in Majlis Sukan Pahang.

Table 4.10 Data Dictionary trip_msp Table.

Field Name	Type	Null
trip_name	Varchar(100)	No
trip_location	vvarchar(50)	No
trip_driver	vvarchar(50)	No
trip_vehicle	vvarchar(10)	No
trip_vtype	vvarchar(5)	No
trip_day	float	No
trip_time	float	No
end_day	float	No
end_time	float	No
trip_participant	int(11)	No

4.3.2 Verification

Verification is a step to make sure that the user inserts the correct data so that there is no error when the data is saved into the database. This is a pre-condition function that checks the input type before the form is submit for saving in the database.

```

if($d_ic_no == "")
{
echo "<script type = 'text/javascript'> window.alert('Please enter driver IC
number.')

```

Figure 4.9 Sample verification code to verify the valid IC number input.

4.3.3 Timetable Coding Structure

In this section, the trip added will be displayed from the designed timetable structure. The “While” statement is used to display the data of a trip from the database by matching the day and time. \$j is the day being loop using “for” statement.


```

<tr>
<th>
    1:00
</th>
<?php
    for ($j=1;$j<8;$j++) {
?>
<td style="padding-top:0px; vertical-align:top">
<?php
$qq="SELECT * FROM $dep where trip_day = '$j' && trip_time = '1'";
$result1=mysql_query($q,$conn);

    while($row= mysql_fetch_array($result1))
{
    $tname = $row["trip_name"];
    $tloc = $row["trip_location"];
    $tdri = $row["trip_driver"];
    $tvi = $row["trip_vehicle"];
    $ttype = $row["trip_vtype"];

    echo "$tloc";
    echo "<br>";
    echo "$tdri";
    echo "<br>";
    echo "$ttype - $tvi";
    echo "<br>";
    echo "<hr>";
}
?>
</td>
<?php
}
?>
</tr>

```

Figure 4.10 Sample code for 1.00 timeslot that loop from Monday to Sunday.

4.3.4 Particle Swarm Optimization (PSO) Algorithms

This section consists of the fitness calculation of Particle Swarm Optimization (PSO) method. The particles that are mentioned in this technique are driver or vehicle. During the fitness calculation, the hard constraint is considered which is clash free in term of selection of particle.

4.3.4.1 Hard Constraint

In my case, the availability of driver or vehicle will be the hard constraint as there is impossible for a driver and vehicle to work on two different trips.

```

while($row = mysql_fetch_array($result))
{
$state = $row["status"];
if($state=="Yes")
{
    $pfBest[$i] = $row["position"]; //retrieve particle's best location;
    $velocity[$i] = $row["velocity"];
    $dri[$i] = $row["d_name"];
    $i++;
}
else
{    $num_of_particle--;    }
}

```

Figure 4.11 Sample code for checking availability of driver.

4.3.4.2 Implementation of Particle Swarm Optimization Algorithms

By applying Particle Swarm Optimization, driver or vehicle (particle) is chosen randomly after executing the code shown in Figure 4.12.

```

if($i != 0)
{
$myFile = "driver.txt";
$fh = fopen($myFile, 'w') or die("can't open file");
for($x=0 ; $x<$num_of_particle; $x++)
{
    $p[$x] = rand(0,$num_of_particle-1);
    $stringData = "[".$x."]=".$p[$x]."\r\n";
    fwrite($fh, $stringData);
    $pick_dri[$x] = $dri[$p[$x]];
    $stringData = $pick_dri[$x]."\r\n";
    fwrite($fh, $stringData);
    $v[$x] = $velocity[$x];
}
fclose($fh);
}

```

Figure 4.12 Sample code of choose driver randomly.

After the particle (driver or vehicle) is chosen, the particle fitness will be calculated by executing the code shown in Figure 4.13.

```

for($iteration=0; $iteration < $number_of_iteration; $iteration++)
{
for ($x=0; $x<$num_of_particle; $x++)
{
$fitness=50-($p[$x]-5)*($p[$x]-5);
if ($fitness > $pfBest[$x])
{
$pfBest[$x]=$fitness;
$update_query = "UPDATE $driver SET pfBest = '$pfBest[$x]' WHERE d_name
='$pick_dri[$x]'";
$result1 = mysql_query($update_query, $conn) or die("Could not execute query in
dri_pso.php");
$pBest[$x]=$p[$x];
}
if ($fitness > $gfBest)
{
$gfBest=$fitness;
$gBest=$p[$x];
}
}
}

```

Figure 4.13 Sample code of calculating fitness for each driver

From the calculation of fitness for each particle (driver or vehicle) above, the velocity of each particle will also be calculated as shown in Figure 4.14.

```

for ($x=0; $x<$num_of_particle; $x++)
{
    $v[$x] = $v[$x] + $c1 * rand(0,1) * ($pBest[$x] - $p[$x]) + $c2 * rand(0,1) * ($gBest -
    $p[$x]);
    if ($v[$x] > $v_max)
    { $v[$x] = $v_max - $alpha*rand(0,1); }
    if ($v[$x] < $v_min)
    { $v[$x] = $v_min + $alpha* rand(0,1); }

    $p[$x] = $p[$x] + $v[$x] ;
    $update_query3 = "UPDATE $driver SET position = '$p[$x]' WHERE d_name =
    '$pick_dri[$x]'";
    $result3 = mysql_query($update_query3, $conn) or die("Could not execute query in
    dri_pso.php");

    if ($p[$x]<$x_min) { $p[$x]=$x_min+rand(0,6);}
    if ($p[$x]>$x_max) { $p[$x]=$x_max-rand(0,6);}
}
return $pick_dri[$gBest];
}

```

Figure 4.14 Sample Code of calculating velocity for each driver.

4.4 Conclusion

This chapter discuss about the design and implementation phase of the system. In this chapter, the algorithms of Particle Swarm Optimization (PSO) method have been reviewed. Besides, the usage of database is also important as the important information of this system will be saved so that it eases the user to trace back. During the implementation of PSO algorithms, it also will save the best fitness of the particle (driver or vehicle) into the database. As a result, the usability of the system will be enhanced with the appropriate use of database.

Chapter V

Result and Discussion

5.1 Introduction

In this chapter, the result of the system will be tabulated in a timetable so that the user is able to see every in a timeslot. Besides the important thing to be discussed is for every trip, the driver or vehicle that assigned does not repeat for every trip. The advantage for the system is that user no needs to arrange the timetable manually as it will consume time compare to this system.

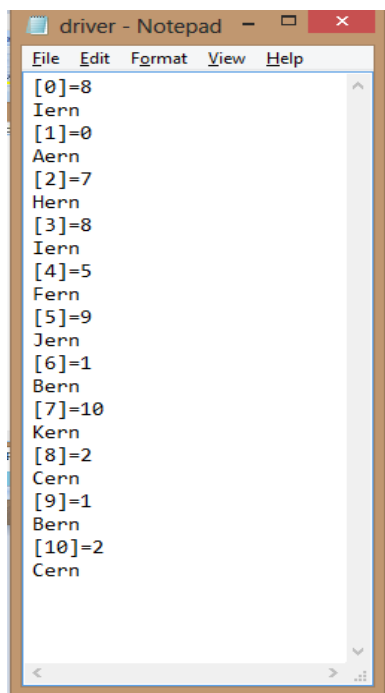
5.2 Result and Discussions

This section will display about the result of the timetable which does not clash in term of driver or vehicle. Figure 5.1 shows the result of timetable for a particular timeslot which does not repeat of driver name or vehicle plate number.

Bahagian Khidmat Pengurusan							
Vehicle	Driver	Trip	Timetable	Department	Logout		
Hours/Days	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
1:00	Kuala Lumpur Dem Bus - CBC 3033						
	Kuala Lumpur Aem Bus - CBB 1236						
	Kuala Lumpur Em Bus - CBB 2213						
	Malacca Fem Bus - CGV 4332						
	Perak Jem Bus - CBB 1122						
	Kuala Lumpur Jem Bus - CDF 3324						
	Johor Bem Bus - CBB 5488						
	Kuala Lumpur Hem Bus - CVD 2211						
	Malacca Gem Bus - CBA 3421						
	Selangor Cem						

Figure 5.1 Arrangement of trip which does not clash in a particular timeslot.

During the generation of timetable, the selected driver is being saved in the text file so that it eases the user to check back which driver is selected during the execution of the Particle Swarm Optimization (PSO) algorithms. Figure 5.2 shows a list of driver that is being chosen randomly from the database in order to be assigned to a trip.



```

[0]=8
Iern
[1]=0
Aern
[2]=7
Hern
[3]=8
Iern
[4]=5
Fern
[5]=9
Jern
[6]=1
Bern
[7]=10
Kern
[8]=2
Cern
[9]=1
Bern
[10]=2
Cern

```

Figure 5.2 List of driver that has been chosen randomly.

Table 5.1 is the tabulated data that converted from the data shown in Figure 5.2. By knowing the position in the array, we will be able to know that which driver will be selected from the list of driver.

Table 5.1 Tabulated list of random driver selected.

Array position	Random Number Generated	Driver at this position referring to database.
[0]	8	Iern
[1]	0	Aern
[2]	7	Hern
[3]	8	Iern
[4]	5	Fern
[5]	9	Jern
[6]	1	Bern
[7]	10	Kern
[8]	2	Cern
[9]	1	Bern
[10]	2	Cern

In order to deal with emergency that might be happened, this system provide a regenerate function so that if a driver takes an emergency leave, the user can assigned a new driver by clicking the “Regenerate” button as shown in Figure 5.2. Figure 5.2 is the display of trip before the “Regenerate” button is pressed.

Bahagian Khidmat Pengurusan									
Vehicle	Driver	Trip	Timetable	Department	Logout				
Trip	Location	Driver	Vehicle	Type of Vehicle	Day	Time	Participant		
IBM Talk	Kuala Lumpur	Dern	CBC 3033	Bus	1	1	40	Cancel	Regenerate
Japanese Show	Kuala Lumpur	Aern	CBB 1236	Bus	1	1	40	Cancel	Regenerate
Japanese Show	Kuala Lumpur	Ern	CBB 2213	Bus	1	1	40	Cancel	Regenerate
Japanese Show	Malacca	Fern	CGV 4332	Bus	1	1	40	Cancel	Regenerate
IBM Talk	Perak	Iern	CBB 1122	Bus	1	1	40	Cancel	Regenerate
IBM Talk	Kuala Lumpur	Jern	CDF 3324	Bus	1	1	40	Cancel	Regenerate
IBM Talk	Johor	Bern	CBB 5488	Bus	1	1	35	Cancel	Regenerate
SQA Talk	Kuala Lumpur	Hern	CVD 2211	Bus	1	1	40	Cancel	Regenerate
SQA Talk	Malacca	Gern	CBA 3421	Bus	1	1	40	Cancel	Regenerate
Software Testing Talk	Selangor	Cern	CBB 3433	Bus	1	1	40	Cancel	Regenerate

Figure 5.3 Detail of trip before the trip is regenerated.

Figure 5.3 below shows the “Regenerate” button is pressed by the user for the last rows of trip which change from the driver name of “Cern” to “Kern” which means that the driver is successfully replaced.

Bahagian Khidmat Pengurusan									
Vehicle	Driver	Trip	Timetable	Department	Logout				
Trip	Location	Driver	Vehicle	Type of Vehicle	Day	Time	Participant		
IBM Talk	Kuala Lumpur	Dern	CBC 3033	Bus	1	1	40	Cancel	Regenerate
Japanese Show	Kuala Lumpur	Aern	CBB 1236	Bus	1	1	40	Cancel	Regenerate
Japanese Show	Kuala Lumpur	Ern	CBB 2213	Bus	1	1	40	Cancel	Regenerate
Japanese Show	Malacca	Fern	CGV 4332	Bus	1	1	40	Cancel	Regenerate
IBM Talk	Perak	Iern	CBB 1122	Bus	1	1	40	Cancel	Regenerate
IBM Talk	Kuala Lumpur	Jern	CDF 3324	Bus	1	1	40	Cancel	Regenerate
IBM Talk	Johor	Bern	CBB 5488	Bus	1	1	35	Cancel	Regenerate
SQA Talk	Kuala Lumpur	Hern	CVD 2211	Bus	1	1	40	Cancel	Regenerate
SQA Talk	Malacca	Gern	CBA 3421	Bus	1	1	40	Cancel	Regenerate
Software Testing Talk	Selangor	Kern	CBV 3422	Bus	1	1	40	Cancel	Regenerate

Figure 5.4 Detail of trip after the trip is regenerated.

5.3 Advantage

For this system, it will benefit the user so that the timetable for each trip can be generated automatically once the trip is added. Besides, during the arrangement of timetable, it will consider the availability of driver and vehicle so that it does not clash among the trip that is added. In addition, this system can handle the emergency that may happen in this timetabling arrangement. The emergency in this case is the emergency leave of driver or break down of vehicle. In case any of these happens, the user is able to regenerate the timetable again so that a new driver and vehicle can be assigned for the trip.

5.4 Limitation

In this timetabling system, there are some limitation that make the system not very user friendly. One of the limitations is that the arrangement of the timetable is being made weekly which means that the schedule will be changed weekly. Besides, the timetable is arranged without consider the date of the trip, it consider for just the day and time of the trip which make the timetable limit to just day and time.

5.5 Contribution

Timetabling system can an issue when it comes to generating a clash free timetable. It will be difficult to arrange as it need to consider the availability of the driver or vehicle. Besides, it needs to consider for every driver or vehicle in a trip in order to choose a suitable driver or vehicle into a trip. It is time consuming to choose driver or vehicle one by one and thus this system will choose the driver or vehicle randomly from the database after considering their availability. Therefore, by applying Particle Swarm Optimization (PSO) method, the driver and vehicle will be chosen by calculating the fitness for every particle, in my case, the particle represents driver or vehicle. After that, the fitness that is calculated which is nearest to 50 will be assigned to the particular trip. Next, the trip that is well arranged will be tabulated and shown in a timetable in order to ease the user to check according to timeslot.

However, this system enable user to handle emergency on driver or vehicle. If in case the driver would like to take an emergency leave, then the user can regenerate the timetable again so that another driver can be assigned to replace the original driver.

5.6 System Testing

This section will performs variety of manual testing on the system. Functional testing is carried out depend on the requirements of the system. Refer Appendix C to view the manual functional test.

5.7 Conclusion

As a result, the timetabling system plays an important role in a transport scheduling system. It should not have any clash incident so that the trip can be held successfully. Besides, in timetable scheduling system, the timeslot is important as it decides which day and what time the trip start and it should not have any mistakes so that it does not delay the trip.

Chapter VI

Conclusion

6.1 Introduction

In this transport scheduling system, the proposed method Particle Swarm Optimization (PSO) method is expected to search for an optimum fitness value for the particle. In my study, the particle mentioned is the driver or vehicle in the search space. By having each particle with its own fitness value, it is easy to search the most optimum position so that it can be assigned for a trip in a department. The technique used for transport scheduling system is appropriate as it choose the available driver or vehicle randomly. Hence, this thesis provide the expected outcome that is optimum and is able to minimize the search time for driver and vehicle which make this techniques more efficient in term of search time.

6.2 Result Analysis

The development of this transport scheduling system, the result of using Particle Swarm Optimization (PSO) has achieved the objectives mention below:

- i. To avoid resource clashing in the same timeslot.
- ii. To choose the optimum fitness value after randomly choose the available driver or vehicle.
- iii. To develop a web-based system for transport scheduling system using PSO method.

6.3 Future Work

In the future of this study, the scope of the task can be widened in which the date of the trip should be considered and the weekly schedule should extend to monthly schedule. Since the scope of the task is widened, the soft constraints should be increased so that it can be more flexible and sensitive to achieve a more optimum fitness value. Besides, the functionality of the system should also be enhanced in order to make the usability of the system more efficient.

6.4 Conclusion

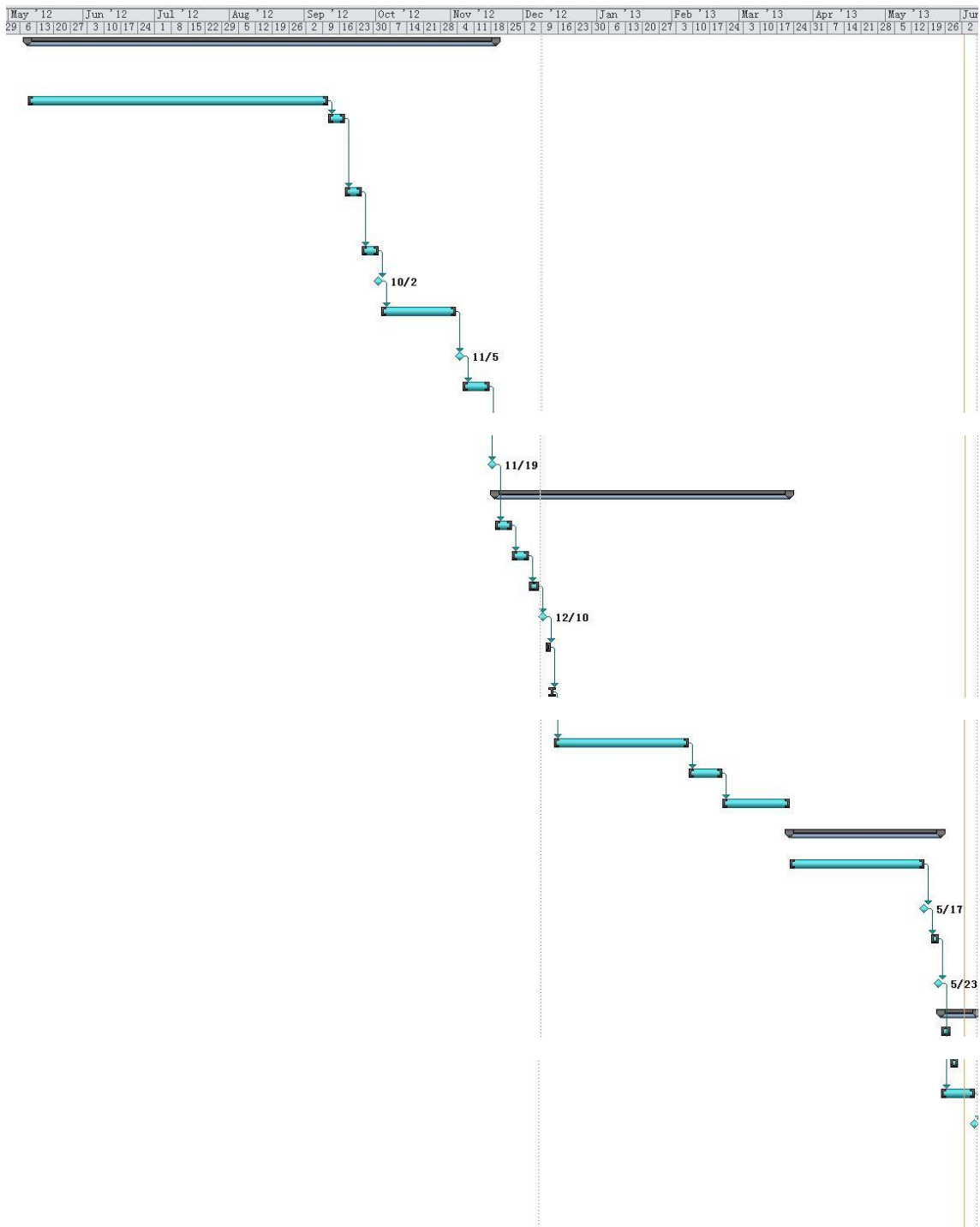
The Particle Swarm Optimization (PSO) method can fulfill the hard constraint which is clash free in terms of selection of driver or vehicle. By filtering the available driver or vehicle from the database, this method can choose randomly from the selected driver or vehicle so that the selection made does not consider every driver or vehicle in the database. After that, the selected driver or vehicle will possess a position value which will be used to calculate the fitness value by applying PSO algorithms. Therefore, the transport scheduling system can be solved by choosing the optimum value of the driver or vehicle.

REFERENCE

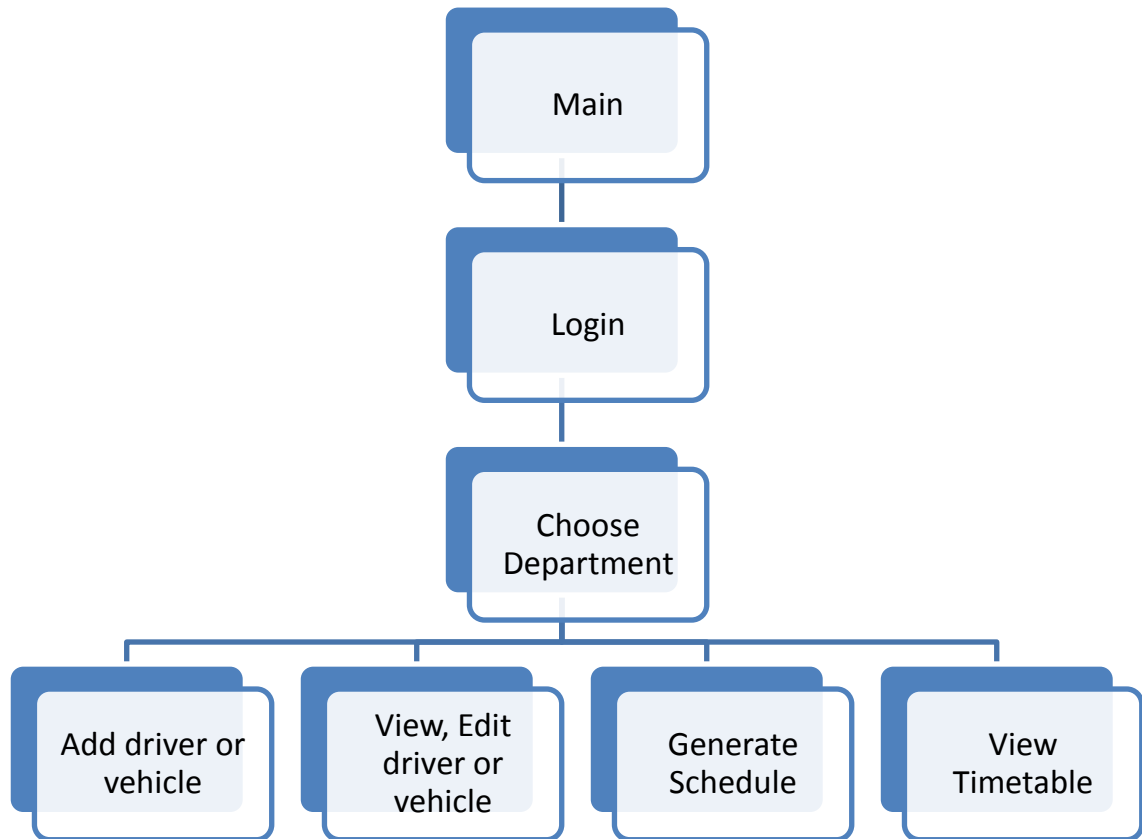
- [1] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, volume 4, pages 1942–1948 vol.4, 1995.
- [2] R. C. Eberhart, Y. H. Shi. (2001). Particle Swarm Optimization: Developments, Applications and Resources.
- [3] Burke, E. K., Kingston, J. H., & de Werra, D. (2004d). Applications to timetabling. In J. Gross & J. Yellen (Eds), *The handbook of graph theory* (pp. 445-474). London: Chapman Hall/CRC.
- [4] Easton, K., Nemhauser, G., & Trick, M. (2004). Sport scheduling. In J. Leung (Ed.), *Handbook of scheduling: algorithms, models, and performance analysis*. Boca Raton: CRC Press, Chap. 52.
- [5] Kwan, R. (2004). Bus and train scheduling. In J. Leung (Ed.), *Handbook of scheduling: algorithms, models, and performance*. Boca Raton: CRC Press, Chap 51.
- [6] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, volume 4, pages 1942–1948 vol.4, 1995.

Appendix A: Project Gantt Chart

	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names
1		1.0 Project Planning and Requirement Analysis	139 days	Wed 5/9/12	Mon 11/19/12		
2		1.1 Get The Title	89 days	Wed 5/9/12	Mon 9/10/12		
3		1.2 Identify Problem Statement, Objective and Scope	5 days	Tue 9/11/12	Mon 9/17/12	2	
4		1.3 Proposal Submission to Research Methodology class	5 days	Tue 9/18/12	Mon 9/24/12	3	
5		1.4 Gather Required Data	5 days	Tue 9/25/12	Mon 10/1/12	4	
6		1.5 Chapter 1 Submission	0 days	Tue 10/2/12	Tue 10/2/12	5	
7		1.6 Study the Literature and Journal	23 days	Wed 10/3/12	Fri 11/2/12	6	
8		1.7 Chapter 2 Submission	0 days	Mon 11/5/12	Mon 11/5/12	7	
9		1.8 Gather Information about the Algorithm and Methodology	9 days	Tue 11/6/12	Fri 11/16/12	8	
10		1.9 Chapter 3 Submission	0 days	Mon 11/19/12	Mon 11/19/12	9	
11		2.0 Iterative Development	88 days	Tue 11/20/12	Thu 3/21/13		
12		2.1 Prototype Design	5 days	Tue 11/20/12	Mon 11/26/12	10	
13		2.2 Prototype Build	5 days	Tue 11/27/12	Mon 12/3/12	12	
14		2.3 Repair Chapter 1 - 3	4 days	Tue 12/4/12	Fri 12/7/12	13	
15		2.4 Chapter 1 - 3 submission	30 days	Mon 12/10/12	Mon 12/10/12	14	
16		2.5 Preparation for PSM I Presentation	2 days	Tue 12/11/12	Wed 12/12/12	15	
17		2.6 PSM I Presentation	1 day	Thu 12/13/12	Thu 12/13/12	16	
18		2.7 Study the algorithm	40 days	Fri 12/14/12	Thu 2/7/13	17	
19		2.8 Preparation of Chapter 4	10 days	Fri 2/8/13	Thu 2/21/13	18	
20		2.9 Code the algorithm	20 days	Fri 2/22/13	Thu 3/21/13	19	
21		3.0 Implementation	45 days	Fri 3/22/13	Thu 5/23/13		
22		3.1 Implementation of coding in system	40 days	Fri 3/22/13	Thu 5/16/13		
23		3.2 Submission of Chapter 4	0 days	Fri 5/17/13	Fri 5/17/13	22	
24		3.3 Preparation Results and Discussion	3 days	Mon 5/20/13	Wed 5/22/13	23	
25		3.4 Submission of Chapter 5	0 days	Thu 5/23/13	Thu 5/23/13	24	
26		4.0 Conclusion	11 days	Fri 5/24/13	Fri 6/7/13		
27		4.1 Combine system	2 days	Fri 5/24/13	Mon 5/27/13		
28		4.2 PSM II Presentation	3 days	Tue 5/28/13	Thu 5/30/13		
29		4.3 Summarise all chapter	10 days	Fri 5/24/13	Thu 6/6/13	25	
30		4.4 Submission of whole thesis with hardcover binding	0 days	Fri 6/7/13	Fri 6/7/13	29	



Appendix B: Hierarchy Chart for Interface Design



Appendix C: System Testing

System Testing – Manual

Module	Item to be tested	Expected Outcome	Result
Login	Login	User should be able to login to the system.	Success
	Username Verification	User will be notified if the username insert is invalid.	Success
	Password Verification	User will be notified if the password insert is invalid.	Success
Timetable	Bahagian Khidmat Pengurusan Timetable	User should be able to view the particular department correctly.	Success
	Bahagian Teknologi Maklumat Timetable	User should be able to view the particular department correctly.	Success
	Majlis Sukan Pahang Timetable	User should be able to view the particular department correctly.	Success
Schedule	Activity Verification	Activity inserted can be verified correctly.	Success
	Location Verification	Location drop down list can be verified correctly.	Success
	Trip Day Verification	Day drop down list can be verified correctly.	Success
	Trip Time Verification	Time drop down list can be verified correctly.	Success
	Participant Verification	Participant input can be verified as number correctly.	Success
	Timeslot clash	Schedule trip dose not clash.	Success
	Driver selection	Choose the optimum driver fitness.	Success
	Vehicle selection	Choose the optimum vehicle fitness.	Success
Add Driver	Name Verification	User should be able to know the validity of name input.	Success
	Identity Card Number Verification	User should be able to know the validity of identity card number input.	Success
	Gender Verification	User should be able to choose the type of gender with the radio button provided.	Success
	Age Verification	User should be able to know	Success

		the valid input for age.	
	Phone Number Verification	User should be able to know the valid phone number is inserted.	Success
	Data Storage	Added driver should be able to be inserted into the database correctly according to the department.	Success
Add Vehicle	Plate Number Verification	User should be able to know the plate number inserted is valid.	Success
	Vehicle Type Verification	User should be able to know the type of vehicle is selected.	Success
	Data Storage	Added vehicle should be inserted into the database according to the department.	Success
Driver	View Driver	User should be able to view the detail of driver according to the department.	Success
	Edit Driver	User should be able to edit the detail of driver if necessary according to the department.	Success
Vehicle	View Vehicle	User should be able to view the detail of vehicle according to the department.	Success
	Edit Vehicle	User should be able to edit the detail of the vehicle if necessary according to the department.	Success
Trip	View Trip	User should be able to view all the trips added in the specific department.	Success
	Cancel Trip	User should be able to cancel trip in the department if necessary.	Success