

CERTIFICATION OF APPROVAL

Route Optimization System

By

Abdul Hayy bin Zulkifli

A dissertation submitted to the
Business Information Systems Programme
Universiti Teknologi PETRONAS
in partial fulfillment of the requirements for the
BACHELOR OF TECHNOLOGY (Hons)
(BUSINESS INFORMATION SYSTEMS)

Approved by,

(Mr. Justin Dinesh Devaraj)

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK
Semester July 2005

CERTIFICATE OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been taken or done from unspecified sources or persons.

(ABDUL HAYY BIN ZULKIFLI)

ACKNOWLEDGEMENTS

Praise Allah the Almighty and Peace and Salutations be upon our beloved Prophet Muhammad (SAW).

First and foremost I would like to thank my final year project partner, Ainulmardhiyah bt Ibrahim as without her, this project would have not been a reality. I would also like to thank my supervisor, Mr. Justin Dinesh Devaraj for all his support and patience in guiding me in producing this final year project. He provided me with simple instructions and never turned me away when I needed his help in some areas of the project.

I would also like to express my gratitude to the Java lecturers that I sought help from, namely Mr. Anang Hudaya Bin Muhamad Amin, Mrs. Amy Foong Oi Mean, and even Mr. Izzatdin bin Abdul Aziz, whom I consulted regarding this project. Even though he was not too familiar with Java, he still tried to help, and it proved to be invaluable.

Last but not least, I must thank my parents for believing in me, that I could complete this project successfully. Your support is a pillar of strength for me.

Thank you very much.

ABSTRACT

This final year project is focused on creating a route generation application that can find the best route on a real road network. The distinct lack of route optimization systems or route generation system in Malaysia was the motivational factor behind this project. It is a collaborative project between another final year student and me. I will be concentrating on finding the best algorithm for calculating the shortest distance between two points on a digital map. After much research into the many algorithms available, and considering some, including Genetic Algorithm (GA), the author selected Dijkstra's Algorithm (DA). This algorithm is an exact algorithm that is specifically for finding the shortest route in a network. It has been used in other networks beside roads, for instance, pipe networks or railway networks. The method used to develop this project is Rapid Application Development (RAD). Using that process, we have settled on Java as our development platform as we found it able to both implement DA and the GIS component (map display and manipulation). The software that we came up with allows the user to select a point of origin on the map and a destination point and expect the software to highlight the best route on the map as well as output the road names in that route, from the origin to the destination. It will also give the user the total distance from origin to destination. It does not take into account the amount of time that the driver might take to travel that distance. This project is just the first step in creating a more robust route generation system for the Malaysian market. The potential for this project to be propagated on a mobile platform can also be considered due to its Java platform.

TABLE OF CONTENTS

CERTIFICATION OF APPROVAL	i
ACKNOWLEDGEMENTS	iii
ABSTRACT.....	iv
TABLE OF CONTENTS.....	v
LIST OF FIGURES	vii
LIST OF TABLES	viii
ABBREVIATIONS AND NOMENCLATURES	ix
CHAPTER 1: INTRODUCTION	1
1.2. Problem Statement	1
1.2.1. Problem Identification.....	1
1.2.2. Significance of the Project	2
1.3. Objectives and Scope of Study	2
1.3.1. Objectives.....	2
1.3.2. Scope of Study	3
CHAPTER 2: LITERATURE REVIEW AND THEORY	4
2.1 Vehicle routing algorithms.....	4
2.2 Java as a Platform.....	8
2.3 Road condition issues.....	8
CHAPTER 3: METHODOLOGY	9
3.1. Procedure Identification	9
3.2. Tools.....	10
3.2.1. Software	11
3.2.2. Hardware	11
CHAPTER 4: RESULTS AND DISCUSSION	12
4.1 Most Suitable Algorithm.....	12
4.2 Screenshots of Dijkstra’s Algorithm (DA)	13
4.3 Performance using Dijkstra’s Algorithm	15
4.4 Integration between GIS and Java.....	16
4.5 Functions of the Route Optimization System (ROS).....	17
CHAPTER 5: CONCLUSION AND RECOMMENDATIONS	19
5.1 Conclusion	19
5.2 Recommendations.....	20

REFERENCES.....	21
APPENDICES	23

LIST OF FIGURES

Figure 1 Rapid Application Development chart.....	9
Figure 2 Screenshot of Textual Output to User.....	13
Figure 3 Dijkstra's Algorithm (DA) highlighting the shortest path between two nodes.....	14
Figure 4 Some of the chosen route is not highlighted in colour.....	15
Figure 5 Output to text of road-by-road directions from Origin to Destination.....	18

LIST OF TABLES

Table 1 Software required for system development.....	11
Table 2 Minimal Hardware required for system development.....	11

ABBREVIATIONS AND NOMENCLATURES

GIS : Geographical Information System

AI : Artificial Intelligence

RAD : Rapid Application Development

DA : Dijkstra's Algorithm

GA : Genetic Algorithm

ROS : Route Optimization System

CHAPTER 1

INTRODUCTION

1.1. Background of Study

Once maps could be digitized, it was only a matter of time that systems were created in order to make use of these maps in order to help humans to navigate the myriad of roads now available. The navigation process is still constantly improving, and this project is yet another step in improving the routing of vehicles from departure to destination. The main difference will be the fact that it takes into account certain obstacles around the way, e.g. whether the road is accident prone or prone to natural disasters.

1.2. Problem Statement

1.2.1. Problem Identification

i. A simple program that can calculate shortest distance between two points that is non-proprietary has not been produced at least in Malaysia. Most of the other software that can calculate this are built-in to commercially available software and so is not automatically available to users. The users will have to purchase the whole packaged software when he/she might only require the shortest distance calculation to be done.

ii. Malaysia is lagging behind in terms of GIS and AI applications. Route navigation systems, for example, have been available in the Western world even before the new millennium. This imbalance needs to be addressed quickly, so as to provide an avenue for computed route use in many fields. For instance, when a hospital

dispatches an ambulance, a computerized route generating system such as this project can help to find the best route.

1.2.2. Significance of the Project

This is a significant project because once a simple route calculation software is available, it can be made commercially available, and users all over Malaysia (at least) can use it to obtain the best route to a destination once suitable digital maps are produced of the areas in question.

There are many areas in Malaysia where a computerized route generation system such as this Route Optimization System (ROS) can be useful. At the very least, it can provide a basis for the route to be chosen in order to minimize travel time and petrol cost. These factors are important to many road users and corporations that deliver products from depot to door, for example.

1.3. Objectives and Scope of Study

1.3.1. Objectives

- i. Provide users with a vehicle route optimization system that can choose a good route from departure to destination.
- ii. Provide an easy to use Graphical User Interface (GUI) that takes into consideration the condition of road bound users.
- iii. Provide a basis for future development in the field of route generation in Malaysia.

1.3.2. Scope of Study

The author will study basically study which algorithm is the most suitable to integrate into the Route Optimization System. Then I will work together with another student who is creating the GIS and GUI part of the System in integrating the components into one cohesive package. The given experimental digital map will only be about 10km by 10km in size. The algorithm used will only compute the shortest path in terms of distance, and time will not be considered at this stage. Directions for navigating from origin to destination will be provided.

CHAPTER 2

LITERATURE REVIEW AND THEORY

A preliminary research was done on several books, journals and articles have given a brief description of vehicle routing algorithm and Java; how it works and also approaches to develop a vehicle routing application on a Java development platform.

2.1 Vehicle routing algorithms

The author also needed to evaluate the best algorithm to use in this specific case. The author came across some journals where the authors have evaluated various algorithms. With specific focus to the Artificial Intelligence component, the author found that some evaluation of using Genetic Algorithm (GA) in route finding was conducted [2]. Overall, GA was found to be effective, especially in a moderately complicated network. This was a stand-alone test.

In another journal, the author looked at GA compared to 2 other algorithms (Dijkstra and Heuristic). Arcview GIS was used and the algorithms were coded using Visual C++, Visual Basic and Mapobject. This test concluded that where small number of nodes and complex conditions are concerned, GA is effective. Dijkstra's Algorithm (DA) favoured a small number of nodes in easy problem conditions [3]. In [3] also, the authors suggested based on their findings, that if the number of nodes is less than 2000, then DA performs better than the rest. They observed that DA performed best if there are a small number of nodes and easier problem conditions.

A good argument for GA is one the author found in a journal from Iran. In that journal from the University of Tehran, Iran, the authors propose that GA is the way to go because a route chosen at the beginning of the journey may not still be the best

route when the user encounters changing road congestion or other issues. Therefore, they argue, the route needs to be re-evaluated constantly, in order to make sure that the user is always on the best route. DA is an exact algorithm that comes up with only one optimal path (unless there is another path with equal lowest cost), and if the user queries it again, it is liable to give the same answer. GA on the other hand, has several solutions in its 'population' and therefore is more ready to give a different solution in case the traffic conditions change for the user [12].

Greedy Random is the name of another algorithm that the author considered. It optimizes the use of Capacitated Vehicle Routing with Time Windows (CVRTW). It is concerned with the famous vehicle routing problem (VRP). VRP is a scenario where there is one depot and several vehicles that have to deliver their load to several clients within a specific time period. The algorithm has to find the best route that the vehicle(s) should follow in order to optimize time and cost.

This approach is commendable, but the problem we are trying to solve in this project does not concern a depot or several vehicles. It is designed with just one vehicle in mind (the user's). However, the time factor in the Greedy Random equation could be useful if it can be adapted for effective use in DA [8].

Ant Colony Optimization is another approach that the author found, however, like Greedy Random, it is specially built to cater for VRP. This particular algorithm can arrive at solutions for business orders known in advance, as well as online (dynamic VRP) where orders arrive during the distribution process [9]. It is concerned with trying to find the best route for a vehicle carrying a specific load for specific customers. Therefore, this algorithm is not suitable for the use of this project as such.

As early as the year 2000, two Cambridge scholars were already trying to come up with a better routing system than the ones available at that time (available in the Western world). They wanted to include traffic conditions into the routing; arguing that a pre-determined route may not be the optimum route once the driver begins the journey, as road congestion conditions always change. Their proposed integrated system included historical road congestion information and a real time component using Global Positioning System (GPS) and Short Message Service (SMS).

They also considered using Dijkstra's Algorithm (DA) in their integrated system, but seeing as our objectives differ from theirs (theirs include a real time component and therefore, time is a factor), we are obliged to go our separate paths. However, they did say that DA guarantees the best route, assuming a route is possible. It is just that, when factoring time into the equation, Fast Lee Routing (FLR), their chosen algorithm worked best [4].

Some scholars in the Beijing Institute of Technology (BIT) came up with the idea of using DA, but with a restricted searching scale, therefore decreasing the time it takes to arrive at the solution. As they illustrated in their figures inside the report, the end result (the optimal path) was the same, but the solution was gained more than 3 times faster. However, we can argue here that Beijing has a much more convoluted system of roads, and a restricted DA would probably be best in that case. We can also note here that despite the many roads that Beijing has, they still chose to use DA, rather than any other type of algorithm, albeit with modifications to the search area to improve the speed in arriving at the solution. This kind of modification can be implemented, if need be, in future, to our system [6].

Another finding is Zhan's paper which is a sequel to another one written by himself, in which he determined that the fastest 3 shortest path algorithms on a real road network are

- i. the graph growth algorithm implemented with two queues,
- ii. the Dijkstra algorithm implemented with approximate buckets (DKA), and
- iii. the Dijkstra algorithm implemented with double buckets (DKD)

In this study, they gave some suggestions as to which Dijkstra algorithm should be used in which situation. We are not considering the graph growth algorithm at this stage. The author suggests that the DKA algorithm is used if the arcs are 1500 or less. If there are more than 1500 arcs, then DKD should be considered [7]. In the case of this project, a generic single source weighted algorithm will be used.

A study was also done specifically on computing one-to-one shortest paths, which is one of the aims of this project. Zhan and Noon produced a paper which pits two high-performance shortest path algorithms against each other, DA implemented with approximate buckets (DIKBA) which is label-setting and Pallotino's Graph Growth Algorithm implemented with two queues (TWO-Q), label-correcting. The authors, in this case conclude that TWO-Q is a better algorithm to use in some situations.

However, they also suggest that the obvious choice of which algorithm to use would be DIKBA assuming the paths are short relative to the extent of the road network [10].

In another journal, a comparative analysis is proposed between C++, Java and ArcView Network Analyst (AVNA)'s implementation of DA. There are differences between the three. Unfortunately, the journal did not detail the results of any tests that were run, if any. The information available is simply what version of DA that each 'contestant' is implementing [13]. This project will help to tackle the comparative analysis.

2.2 Java as a Platform

Another support for our chosen Java development platform is that the two Cambridge scholars also used Java, citing its ability to run on web browsers in homes and offices and nowadays perhaps on mobile phones [4].

The usage of Java as an effective platform is proven by a prototype in Zhejiang University of Finance & Economics, Hangzhou, China, where a Web-GIS tool was created using Java for displaying the GIS component, quoting ubiquitous use on different platforms. It had no problems in displaying GIS file formats [11].

2.3 Road condition issues

It has been proven that incorporating historical traffic data into vehicle navigation systems can lower costs and vehicle usage, especially with regards to commercial vehicles [1]. Taking this into consideration, it would be a good idea to try and implement historical traffic and road condition data into a vehicle navigation system.

CHAPTER 3

METHODOLOGY

3.1. Procedure Identification

The system will be developed using Rapid Application Development (RAD) approach. RAD techniques emphasize quick creation of software and this allows frequent testing of the smaller functionalities in order to create accurate and reliable software for the users.

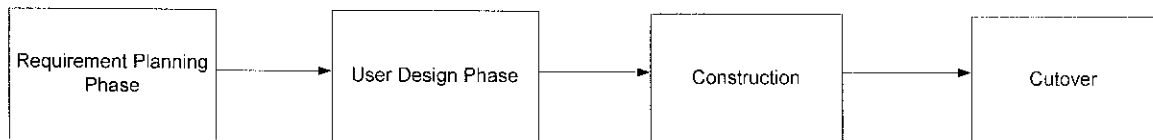


Figure 1: Rapid Application Development chart

- i. Requirement Planning Phase is implemented in this project by ascertaining what functionalities the project should have. The author did this by looking at many journals and articles related to this field of study. There were some examples that the author looked at in the commercial software already available as well. Then, the author discussed with his counterpart in order to come up with respective responsibilities in producing the components for the project that can be integrated into the System. From the functionalities that we choose to have in the software, we know what parts we have to work on and which tools to use.
- ii. User Design Phase is the next step in RAD where we start to flesh out our ideas for the software. Once we have a flow of how the system should work, we can

begin to design the interface. We do this with a generic graphical user interface that can be used easily and intuitively, in mind. For the most part, the algorithm will be transparent to the user, so the most visible part will be the GUI and the digital map.

- iii. Construction follows User Design, because by now we have the idea of how the system should look like and respond to the various user requirements. We then use the programming language chosen to realize the System and test its functionality to ensure it does not have any bugs/errors.
- iv. The next phase, Cutover, is when the System is deemed fully ready for use. It should not have any major errors, and it should fulfill the objective that we set out to achieve at the beginning of the project. Once a normal user has begun using the software, we can get a better idea of where the System is in terms of fulfilling the requirements. Does it need more work to make it more robust/accommodating to other scenarios, for example? These factors can be considered later on, and then we can revisit the previous cycle(s) and work on improving the project from there.

3.2. Tools

The following are the tools suggested to be used in the development of the project:-

3.2.1. Software

No.	Software	Description
1	ESRI® ArcGIS 3.2	For analyzing, mapping, managing, sharing, and publishing geographic information
2	Java 2 Platform Standard Edition 1.5.0	The chosen programming language for this project.
3	Microsoft® Access	For database management
4	Sun Java One Studio (Forte) 4	Integrated Development Environment (IDE) that allows RAD approach to constructing the project

Table 1: Software required for system development

3.2.2. Hardware

No	Device	Requirement
1	Operating System	Microsoft Windows XP
2	Processor	Intel Pentium 4 2.40 GHz
3	Memory	512 MB of memory
4	Disk Space	< 1GB of free space
5	Other Peripherals	Screen, Keyboard, Mouse

Table 2: Minimal Hardware required for system development

CHAPTER 4

RESULTS AND DISCUSSION

This chapter details the current findings and development choice of this project. Some of the information are from journals or articles while others were obtained through trial and error in the project undertaking.

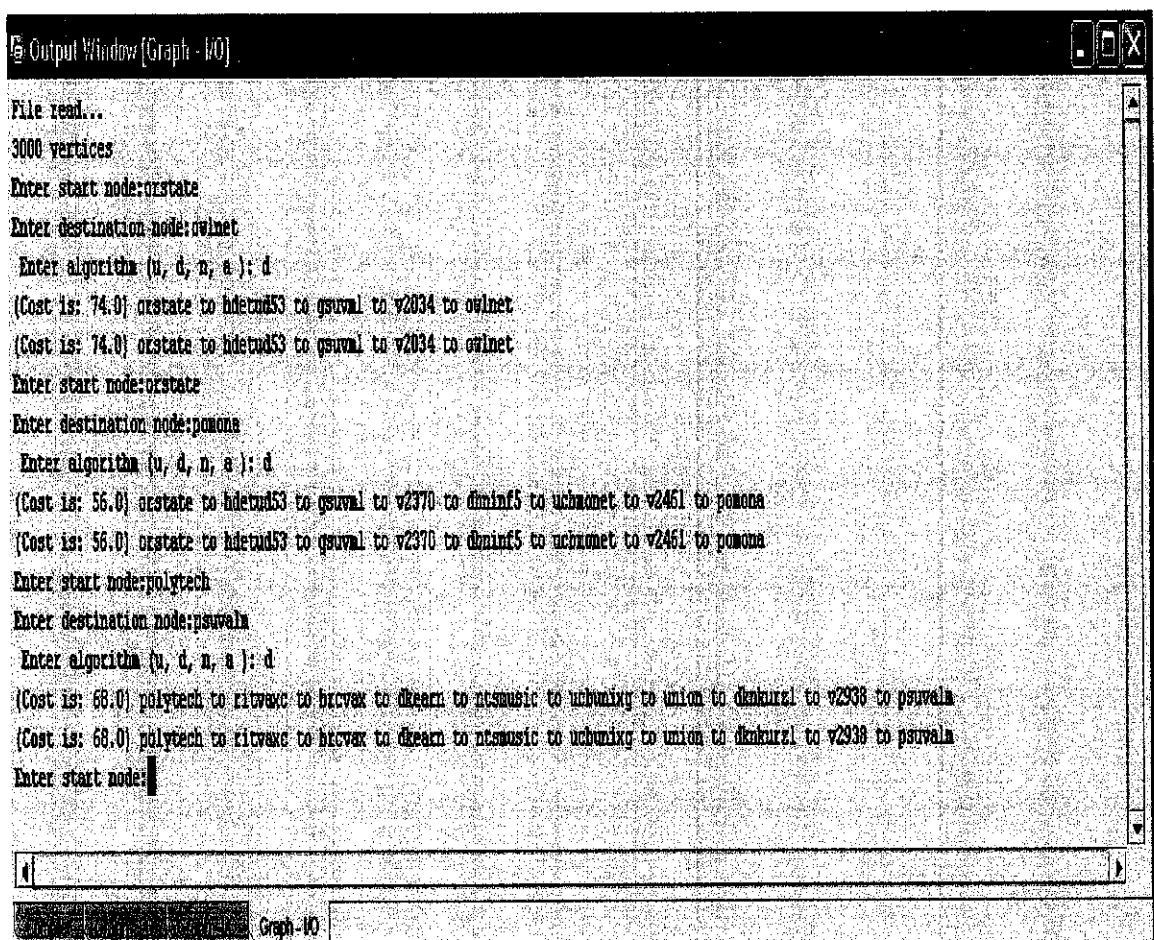
4.1 Most Suitable Algorithm

Initially, the author thought the best algorithm to use would be Genetic Algorithm (GA). And all development platform searches were performed on this basis. However, once the scope of the project was realized, in that, all that is needed is to find the optimum path; the author had to re-evaluate the available algorithms again in order to find the most suitable for the current scope. The most famous algorithm for finding the shortest path (through the lowest cost of the edges) is Dijkstra's Algorithm (DA).

The author had seen several comparisons between DA and GA (though not always exclusively between the two) in route generation reports, and deemed that DA was the simpler path to take, requiring less effort but producing a satisfactory result. GA is a more complex algorithm and has many potential uses in many industries, but DA is certainly more specialized in route optimization and can cater to our needs better. Using GA at this stage would have meant to really understand the wide area that is GA, and then only begin to apply it to the Geographical Information System (GIS) application that we are creating. With DA, the process is more straightforward, and as we can see, it is modifiable to suit different needs. In fact DA exists already in many different forms throughout the industry, some with limited search space, in order to optimize the time taken to arrive at a solution.

4.2 Screenshots of Dijkstra's Algorithm (DA)

In order to show the applied algorithm in this project, a screenshot of the displayed route on the map will not be sufficient (as the process is not shown). However, with the same algorithm, I can show you in textual format, the shortest path.



```
Output Window (Graph - I/O)
File read...
3000 vertices
Enter start node:orzstate
Enter destination node:ovlnet
Enter algorithm (u, d, n, a ): d
(Cost is: 74.0) orzstate to hdetud53 to gsuval to v2034 to ovlnet
(Cost is: 74.0) orzstate to hdetud53 to gsuval to v2034 to ovlnet
Enter start node:orzstate
Enter destination node:pomona
Enter algorithm (u, d, n, a ): d
(Cost is: 56.0) orzstate to hdetud53 to gsuval to v2370 to diminf5 to uchnmet to v2461 to pomona
(Cost is: 56.0) orzstate to hdetud53 to gsuval to v2370 to diminf5 to uchnmet to v2461 to pomona
Enter start node:polytech
Enter destination node:psuval
Enter algorithm (u, d, n, a ): d
(Cost is: 68.0) polytech to ritwak to hcvax to dlearn to ncsmusic to uchning to union to dknuzel to v2938 to psuval
(Cost is: 68.0) polytech to ritwak to hcvax to dlearn to ncsmusic to uchning to union to dknuzel to v2938 to psuval
Enter start node:
```

Figure 2: Screenshot of Textual Output to User

The algorithm was run on a file containing 3000 nodes. This is to approximate the load taken by the algorithm in a real-world situation.

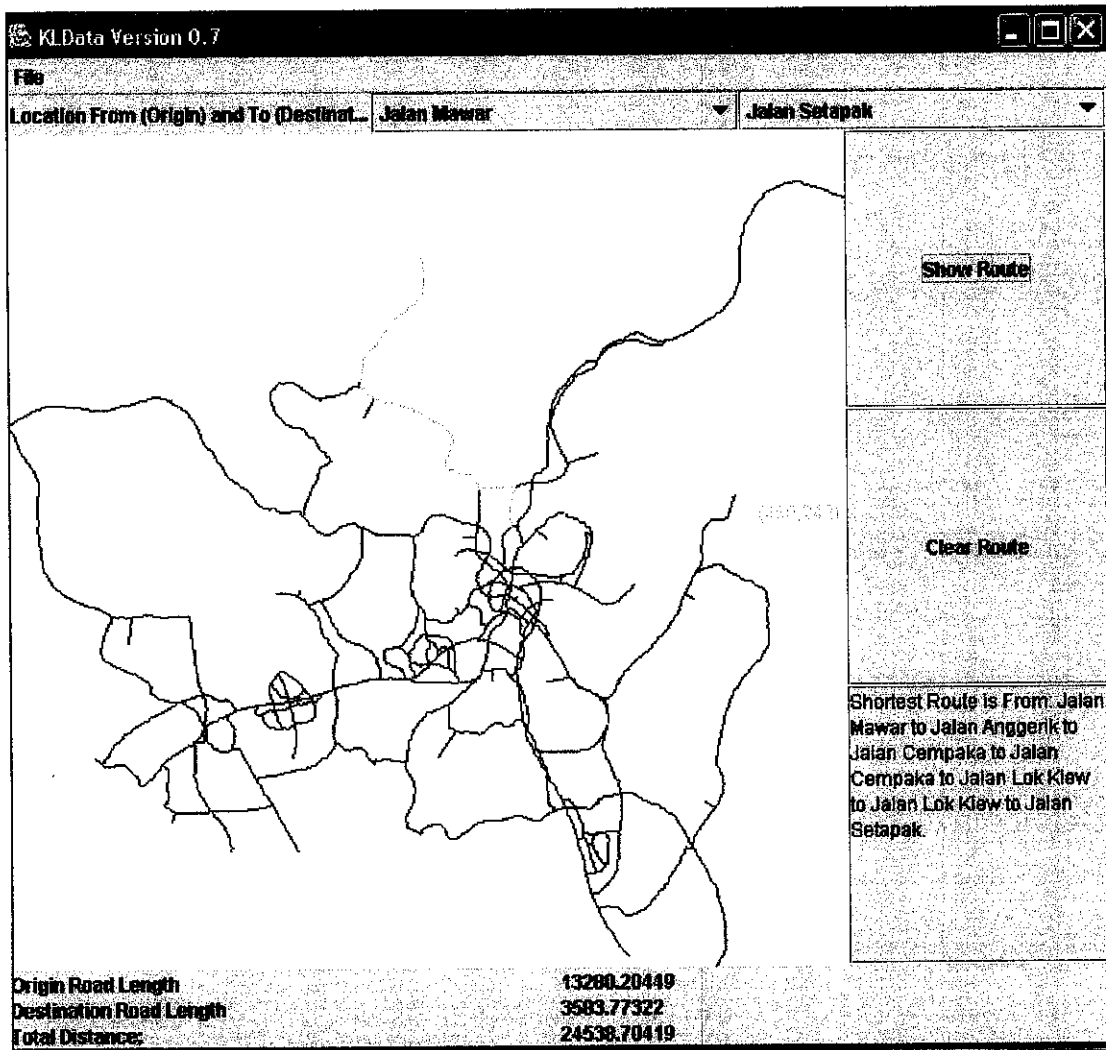


Figure 3: Dijkstra's Algorithm (DA) highlighting the shortest path between two nodes

When the program is run, we can see the highlighted route as above. However, due to the fact that the highlighting was programmed by highlighting the roads used in the shortest route, (a less than ideal solution, admittedly) it sometimes does not highlight the route correctly, as users might not use the full length of the road in their travels. They are likely to turn in to some other road. Below is evidence of the problem:

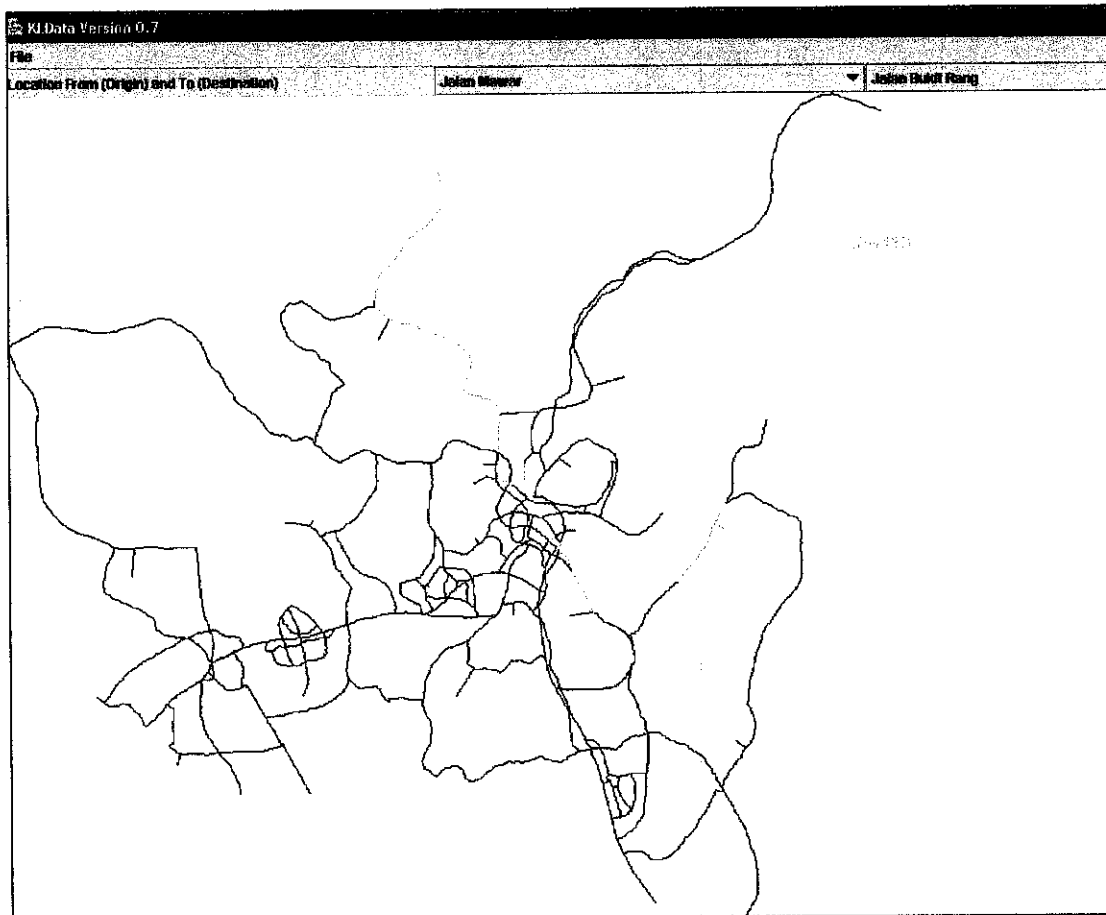


Figure 4: Some of the chosen route is not highlighted in colour

4.3 Performance using Dijkstra's Algorithm

The author found out that ArcView's Network Analyst (AVNA) uses a modified version of DA with a d-heap in combination with a custom memory management to deal with very large networks. In that case, any performance comparison between this project and AVNA would be between different implementations of DA. The algorithm used in this project is a more generic version of DA.

We tested the generic algorithm with the shp file that we are using (road1.shp which has 339 nodes) for this project, and the algorithm gave an answer instantly. We tested AVNA with the same shp file, and it arrived at the solution very quickly, about

a split second after the mouse was clicked. The speed differential between the two is imperceptible, unless a more accurate timing method is used.

The author theorizes that this is due to the small amount of nodes in the test file. At this level, the performances of both DA implementations are, to the normal user who does not have a very accurate timekeeping, absolutely similar. If the test file were bigger, containing more than 5000 nodes, then perhaps there might be a perceptible difference.

The author was able to obtain a text file containing 3000 nodes from Weiss (<http://www.cs.fiu.edu/~weiss/dsj2/code/code.html>) and tested the DA used in the project with it. It also produced the results instantly.

We will test this in the future with AVNA, but for the purpose of this project, there is no difference, and it is already very quick in arriving at the solution, highlighting the success of the chosen algorithm, DA.

4.4 Integration between GIS and Java

The development platform shifted throughout the project from initially, MATLAB, until most recently, and concretely, Java. MATLAB has certain tools to cater for route generation, through its Logistics Engineering Toolbox, but this particular toolbox is a special item and is therefore unavailable initially in any of the University's computers. It is not even sold in the shops. Ordering the toolbox takes too much time, so we had to find another platform for the algorithm.

C++ was the next best choice, but if the author wanted to use C++, which is an older language, it seemed only logical to see what other, more current and widespread language, would be a better candidate. Realizing this, the author looked at the Java platform.

Java is nowadays used to even create mobile phone games. This gives an avenue to developing on the Java platform that C++ does not have. The potential for this project to be propagated (but first adapted) through mobile phones is very interesting.

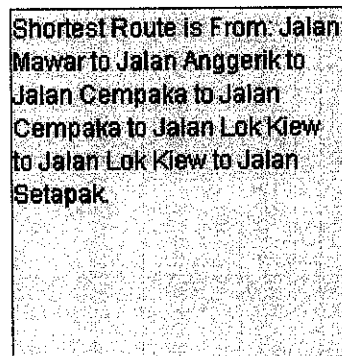
However, more to the point, Java is an object-oriented language, and just by looking through the internet, the author found several examples of DA implemented in Java. This makes the task much simpler, as the algorithm is already proven to run on a Java platform, and all that is required is to integrate with the GIS component.

Even on that front, Java has already been used to develop GIS software. There were several tools available to us (Netmaps, OpenMap are the two we tried), but in the end, adapting them to our project proved somewhat troublesome (lack of documentation, expertise and time to learn the software), therefore, we had to produce an application based on what we know and some coding made available on the internet.

4.5 Functions of the Route Optimization System (ROS)

ROS is able to determine the shortest path on a one-to-one basis. That is, one origin, to one destination. It takes into account the distance as the cost of the edge in order to determine the shortest path, therefore the shortest path is the path with the least distance to travel. The user selects the origin from the pull down menu (combo box),

as well as the destination. Since some users will not be too familiar with map-reading, we included the origin and destination selection in that manner. The user submits the query to the system, and the system responds with the shortest path highlighted on the map. Again, this might not be useful to users who are not able to read maps that well, so a road-by-road instruction set from the origin to the destination is given. An example of the instructions given is in Figure 4 (below).



```
Shortest Route is From: Jalan  
Mawar to Jalan Anggerik to  
Jalan Cempaka to Jalan  
Lok Kiew to Jalan Lok Kiew  
to Jalan Setapak.
```

Figure 5: Output to text of road-by-road directions from Origin to Destination

The project is not able to factor in time factors into the equation. The user cannot input that he/she wants to arrive at a destination within the hour, for example. This calculation requires the system to have the database of the speeds traveled along the various roads on the road network. That is, on Jalan Mawar (for example), the average speed is 80km/h. This is an example of the data required in order to perform this calculation, and factor time into the output.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Conclusion

Although vehicle routing applications are not state-of-the-art any more, the need and therefore, the market for such a product is growing in Malaysia. As we know, Malaysia is a bit behind the Western countries and even some Asian countries, in implementing new technology, and applying vehicle routing systems in Malaysia is no exception.

With the large number of vehicles now on Malaysian roads, it is important that the user is given the information to decide about how best to get to their destination from their point of origin.

With the large car manufacturers starting to make an effort with a Malaysian digital map developer to produce digital maps compatible with their built-in car navigation systems for Malaysian roads in the near future, this has spurred the author and his partner to come up with software that can be a platform for future developers.

In conclusion, creating the Route Optimization System on a Java platform using Dijkstra's Algorithm (DA) results in a very flexible platform due to the Java language, and allows it to be dispersed even across to mobile platforms. DA itself is also easily modifiable to meet different routing needs if need be.

5.2 Recommendations

The author recommends any future work to look into a further optimized DA, as well as a better implementation of road condition information into the system. This is to further enhance the system's capabilities and to ensure it is as accurate as possible. Further future work can be done in the form of integrating real time information, as in [4], integrating computing, communications and location based services to produce the most accurate routing for that particular moment in time.

REFERENCES

- [1] Kim, Seongmoon, Lewis, Mark E., White III, Chelsea C., "Optimal Vehicle Routing with Real-Time Traffic Information", 26, (2004).
- [2] Delavar, M.R., Samadzadegan, F., Pahlavani, P., "A GIS-Assisted Optimal Urban Route Finding Approach Based On Genetic Algorithms", 1-4, (2004).
- [3] Shad, R., Ebadi, H., Ghods, M., "Evaluation of Route Finding Methods in GIS Application", 1-4, (2004).
- [4] Fawcett, J., Robinson, P., "Adaptive Routing for Road Traffic", University of Cambridge, IEEE Computer Graphics and Applications, (2000).
- [5] Moin, N.H., "Hybrid Genetic Algorithms for Vehicle Routing Problems with Time Windows", 15, (2002).
- [6] Mengyin Fu, Jie Li and Zhihong Deng, "A Practical Route planning Algorithm for Vehicle Navigation System", Beijing Institute of Technology (Online), <http://www.paper.edu.cn>, 2004.
- [7] Zhan, F. Benjamin, "Three Fastest Shortest Path Algorithms on Real Road Networks: Data Structures and Procedures", Journal of Geographic Information and Decision Analysis, vol.1, no.1, pp. 69-82, 1997.
- [8] Dominik R. Rabiej, "Greedy Random: A Novel Algorithm for Vehicle Routing Optimization", Intel Science Talent Search, November 29, 2000.
- [9] Luca Maria Gambardella, Andrea E. Rizzolia, Fabrizio Oliverio, Norman Casagrande, Alberto V. Donati, Roberto Montemanni, Enzo Lucibello, "Ant Colony Optimization for vehicle routing in advanced logistics systems", Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA), Switzerland, 2004.
- [10] Zhan, F.B., Noon, C.E., "A Comparison Between Label-Setting and Label-Correcting Algorithms for Computing One-to-One Shortest Paths", Journal of Geographic Information and Decision Analysis, vol. 4, no. 2, 2000, pp. 1-13
- [11] Xiaolin Lu, "A New Approach for Web-GIS Based Collaborative Transportation Planning System Design", Zhejiang University of Finance & Economics, Hangzhou, China, 2003.

[12] M. R. Delavar, F. Samadzadegan, P. Pahlavani, “*A GIS-Assisted Optimal Urban Route Finding Approach Based On Genetic Algorithms*”, University of Tehran, Tehran, Iran, 2000.

[13] Ross Sherlock, Peter Mooney, Adam Winstanley, Jan Husdal “*Shortest Path Computation: A Comparative Analysis*”, Department of Geography, University of Utah, 1997

APPENDICES

- Appendix 1 Intuition behind DA
- Appendix 2 DA illustration
- Appendix 3 Running DA example
- Appendix 4 End of running DA
- Appendix 5 Main DA coding modified from Weiss source codes

Intuition behind Dijkstra's algorithm

Imagine that each node is a small weight, and that each edge is a string of the appropriate length connecting these weights.

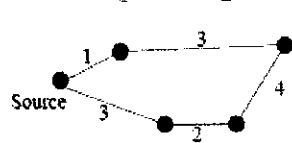
Initially, entire contraption is on a table.

Pick up the source node from the table and keep lifting it until all nodes are off the table.

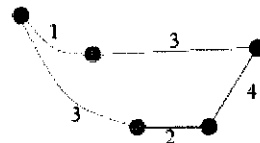
Height of source when a node comes off the table = shortest distance from source to that node!

Appendix 1: Intuition behind DA

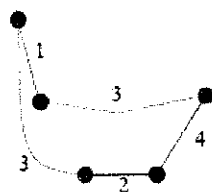
Analog computer for Dijkstra's algorithm



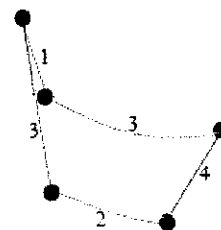
(a) Initial Configuration



(b) Source node is lifted



(c) Two nodes are lifted

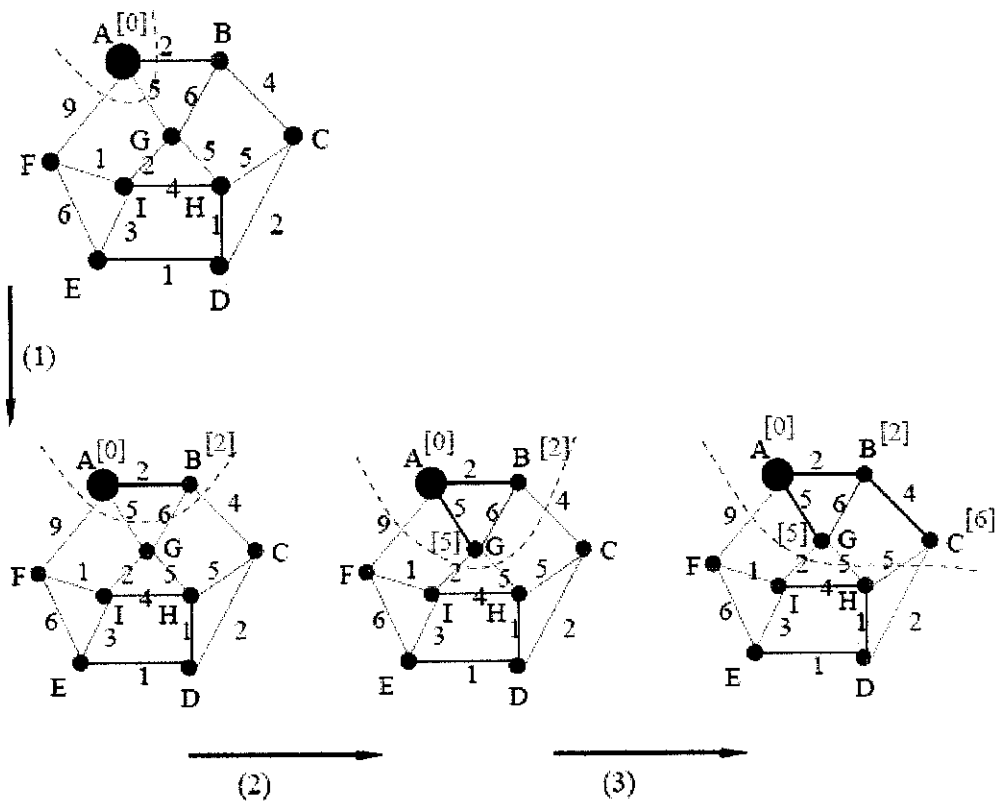


(d) Three nodes lifted

Blue node: lifted, Red node: not lifted, but connected to a blue node by some edge
 Black: all other nodes Bridge: edge between blue and red node

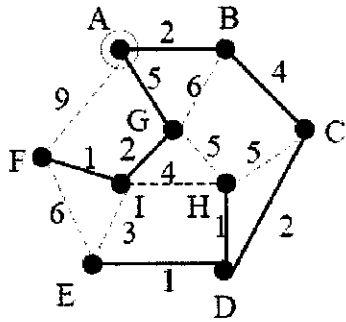
Appendix 2: DA illustration

A few steps of Dijkstra's algorithm for running example



Appendix 3: Running DA example

End of Dijkstra's algorithm



Edges on shortest-path routes are thick.
These edges form a tree.

Appendix 4: End of running DA