DJIKSTRA ALGORITHM BASED APPROACH TO SHORTEST PATH MODEL IN PUBLIC BUS TRANSPORTATION

DEANDRHA AGUSTA & FRISKA NATALIA FERDINAND

Department of Information System, Universitas Multimedia Nusantara, Indonesia

ABSTRACT

In these days, as the android is getting more popular, lots of Android applications that are used for the public interest and attracts much more attention. Mobile applications are a rapidly growing segment of the global mobile market and the mobile application is made to solve problems in everyday life. This study is designed to help the users of public transport, in the search for the bus stop and the route to be taken to get the bus stop. Google Maps is one of the applications, which can help users in solving problems that in the search location. However, the data existing locations on Google Maps for public bus stop, such as kopaja and metromini are still not available and also, the shortest route is not provided. Based on these problems, it is necessary to search Android apps bus service, solely devoted to the type of public bus transportation is bus types kopaja and metromini. Results from this study are an application, that can help users find the shortest route and give recommendations on the bus type (Kopaja or Metromini). The mobile application developed in this research, which can make all users can be easier to find the bus stop and get the shortest route to the bus stop. A solution heuristic, based on a DJIKSTRA algorithm is also suggested. Its performance is evaluated through an example problem.

KEYWORDS: DJIKSTRA Algorithm, Shortest Path & Public Bus Transportation

Received: Sep 28, 2017; **Accepted:** Oct 19, 2017; **Published:** Nov 13, 2017; **Paper Id.:** IJCSEITRDEC20171

INTRODUCTION

Generally, citizens do not make full use of public transportation system in Jakarta, because the citizens are on lack of the data and information. Most of Jakarta citizens still do not pay enough attention to the public transportation, due to the several issues, such as the insufficient data and information of the public transportation, the confident in the safety of the public transportation and the lacking of the comfortable, and speed of the public transportation. Mobile applications, usually help users to connect to the Internet service that is commonly accessible on PC, or facilitate them to use internet applications on the device, that can be carried (Turban, 2012).

This study proposes an alternative to the problem of finding the optimal path, in public transportation, in order to help the users of public transport, in the search for the bus stop and the route to be taken to get the bus stop. In search of the bus stop, the user can detect its position using GPS, to find the nearest bus stop from their location. Google Maps is one of the applications that help users to search location. But, the data existing locations on Google Maps, for public bus stop as kopaja and metromini not available, and the service that is given is not the shortest route. Based on these problems, it is necessary to search Android apps bus stop location, that is solely devoted to the type of public bus transportation, is bus types *kopaja* and *metromini*. The main problem is bus queuing, due to slow boarding times, also caused by weak schedule enforcement.

There have been some studies, related to DIJKSTRA algorithm. Budiarsyah (2010), Dijkstra's algorithm is faster to use, than the Bellman-Ford algorithm and Floyd-Warshall. Ni kai (2014) provided a web application, for finding optimal routes from locations of specialized response team stations to incidents site, so as to maximize their ability to respond to hazard incidents. Dijkstra's algorithm is a greedy algorithm, that used to solve the problem within the shortest path problem, for a directed graph with edge weights valued non- negative (Thomas, 2009). Also, there is some study discussion, about the method to designs the application, which is prototype model. (O'Brien, 2005), Prototype method is suitable to create software, that can be changed in accordance with the demands and needs. Therefore, the purpose in this study to develop the mobile application, based on the Dijkstra's algorithm as the principle of priority queues. The route to be searched by Dijkstra's algorithm is the route that has the highest priority, only. In determining the priority routes or knot this, Dijkstra's algorithm compares each value or node weights, that are in one level. Then, the value or weight of the node is stored for comparison, with the value of the new route will be found.

The remainder of the paper is as follows. The problem statement and previous studies, related to the problem considered are introduced in Section 2. Section 3 describes definition of the problem and the solution procedure. In Section 4, a numerical example is performed with the solutions. Finally, the conclusions and further research areas, appear in Section 5.

LITERATURE REVIEW& PROBLEM STATEMENT

This study investigates the shortest path in public transportation network, among two types of bus: Kopaja and Metromini, in Jakarta Barat Region. Based on this purpose, the pseudocode in DIJKSTRA algorithm was developed in three kind of nodes. There are three types of nodes, that describe the status of each node is traced (Edmonds, 2008):

- The node or nodes that have been discovered and yet to be addressed.
- Node or node that has not been found but have not been addressed.
- The node or nodes that have been found and has been handled.

Node is a node, that is formed of two dots that form a pathway. In order to create a node graph maker, tools needed to turn the node into a coordinate point, the Google Maps API is used to retrieve coordinates and by using this tool, it will be faster to use, than take the coordinates manually using Google Maps. There is a red marker, which is the node that is used to mark the beginning and end of every stripe, a green marker, which is the location of the bus stop and a collection of blue circle, which is the path that has been made. Figure 1, describes the coordinate data, which is obtained by the steps that have been done before.

- Setting the initial node to node or status has been found (found) and addressed (handled).
- Perform a search on each node or nodes, that possibility can be reached directly from the node or nodes, that are being visited. If the points obtained has never been discovered or addressed beforehand, then his status changed to found (founded). If the points obtained have been found or previously dealt with, then the value of the point is updated based on the value or weight of the smallest.
- Conduct a search of the point which has the smallest value of each node with the status found and then visit the knot.
- Perform repetitions in a sequence based on the above steps until all the nodes are found.

The used method in the design of this application is the prototype method. Prototype method is a method that presents a complete overview of the system, the user can view the system modeling of the side view as well as procedural techniques that will be built (Kendal, 2003). The purpose of using the prototype method is to perform the initial software development models into a final software. The first step uses a prototype method is to identify the needs of the user. Identifying the needs of users is done by directly asking the user. Once identified, the next stage is to develop the prototype. If the prototype was well received, then the prototype can be used to design the system in the next stage of the final stage.

RESULT AND ANALYSIS

This section describes the solution procedure based on DJIKSTRA Algorithm, using an illustrative example. In conducting the search path, Dijkstra's algorithm search across all available paths, and then chooses the shortest path, among all paths have been traced. In its application, the search is done is finding the shortest route to the bus stop in the area of West Jakarta. The system will be designed to cater, for users of Android smartphones, to find the bus stop in accordance with routes. Users can specify the locations you want to target and then, the system will help in the search for the bus stop. The application will show the shortest route, using Dijkstra's algorithm and the type of bus to be ridden by users.

Figure 2: General Overview of the System

Based on Figure 2, the user is required to access the application via smartphone and then, those users make requests to the application. Furthermore, the application reads data from the database information, in accordance with user demand. Database transmit information and data requested in advance via smartphones through to the end users. The Sqlite database is used in the design of this application and the stored data, has been shaped and graph coordinate data path. The table will be created on the Sqlite database are as follows:

- Table vehicle, used to save the data buses, such as the name and number of bus lines are also the route of the bus, using the code node, which is a collection of point, coordinates. Code knot used to assist in reading these applications on a bus route. Eg, code nodes (0-1) which mean that, the bus runs from 0 to the location 1 location.
- Table graph, used to store data paths based on the code node, using the coordinates obtained from Google Maps.
- Table destination, used to store data intended location, using coordinates obtained from Google Maps.

In the design of the application of this study, there is a point whose coordinates are in the form of bus lanes, intersections and stop goals used as a marker of an existing location on Google Maps. The bus line is a collection of point coordinates are used to calculate the distance from one location to another. In conducting the search path, Dijkstra's algorithm search across all available paths and then chooses the shortest path, among all paths have been traced. In its application, the search is done, by finding the shortest route to the bus stop in the area of West Jakarta. There are several bus stops, that became the destination of the shortest route search application shows in Table 1:

Table 1: Data of Bus Stops in the Area of West Jakarta

No	Bus Stop Name	Coordinate
1	Bus Stop JembatanBaru	-6.154592736647674, 106.73060595989227
2	Bus Stop JembatanBesi	-6.152077986002897, 106.79492264986038
3	Bus Stop Pos Grogol	-6.166040968459322, 106.79737150669098
4	Bus Stop Ukrida	-6.17233032399962, 106.78863421082497
5	Bus Stop Durikelapa	-6.1852128029787075, 106.76840096712112
6	Bus Stop Kedoya Green Garden	-6.164478284703887, 106.76275223493576
7	Bus Stop TubagusAngke	-6.152760676644125, 106.77130579948425
8	Bus Stop Taman Harapan Indah	-6.143816293060353, 106.78596138954163
9	Bus Stop JembatanDua	-6.143296266133567, 106.79382562637329
10	Bus Stop Mangga Raya	-6.173472986688517, 106.77805423736572
11	Bus Stop DaanMogot 3	-6.162091583053505, 106.77128970623016
12	Bus Stop Pertamina	-6.166368971461574, 106.7869645357132

Based on the Table, there are three scenarios in the path selection Dijkstra's algorithm:

Scenario 1 – Find the Routes with 2 Lanes

Figure 3: Scenario 1 DJIKSTRA's Algorithm

Based on Figure 3, blue circle is a node, the red circle is the way that has not been traced and a yellow circle is the beginning and end of the route search scenarios, with 2 lanes. Here are the results of the scenario, using the shortest path algorithm Dijkstra:

Table 2: Table Scenario 1

Node	Status	Weight	Route
Start	Handled	0	Start
B	Handled	3743.286	Start – B
C	Discovered	6250.2344	Start – B – C
D	Discovered	7928.2931	Start – B – C – D
E	Handled	4308.0653	Start – B – E
F	Handled	6837.0604	Start – B – E – F
G	Discovered	10825.8612	Start B – C – D – G
Finish	Handled	8895.2734	Start – B – E – F – Finish

Based on Table 2, to reach the destination all the points that are available to be found to compare which path is shorter. A comparison of the path is found, the status "found" changed to "visit", when the weight is on the line is shorter than the path being compared. The shortest route is obtained from the application of Dijkstra's algorithm, in scenario 1 is the start -B- E- F- finish with a total weight of 8895.2734. Figure 4, described the result of the shortest path, taken by the Dijkstra's Algorithm.

Figure 4: Result of Scenario 1 DJIKSTRA's Algorithm

The path taken starting from the start point in the path A - B, then to point B, E, F, and then arrive at the finish point in the path F- G. The line is the shortest path to the total weight of 8895.2734. The shortest path can be found by assuming all lines, have been visited so that, the shortest path can be found.

Scenario 2 – Find the Routes with 3 Lanes

Figure 5: Scenario 2 DJIKSTRA's Algorithm

In Figure 5, the blue circle is the node, red circle is a path that has not been traced, a yellow circle on the path A - B is a starting point and a yellow circle on the path D- G is the end point. Here, is a table of results from a search shortest path:

Table 3: Table Scenario 2

Node	Status	Weight	Route
Start	Handled	0	Start
B	Handled	3743.286	Start – B
C	Discovered	6250.2344	Start – B – C
D	Discovered	7928.2931	Start – B – C – D
E	Handled	4308.0653	Start – B – E
F	Handled	6837.0604	Start – B – E – F
G	Discovered	10825.8612	Start – B – E – G
Finish	Handled	8895.2734	Start – B – C – D – Finish

Based on Table 3, the shortest route gained from the implementation of Dijkstra's algorithm, in scenario 1 is the start - B - C - D - finish with a total weight of 9185.4481. That is much shorter than the point G, with a total weight of 9836.137. To reach the destination, all the points that is available to be found to compare which path is shorter. A comparison of the path is found, the status "found" changed to "visit", when the weight is on the line is shorter than the path being compared. The Figure 6, describes the result of the shortest path taken by the Dijkstra's algorithm:

Figure 6: Result of Scenario 2 Dijkstra's Algorithm

The starting point on the path A- B, marked by a yellow circle, then to point B, C, D, and then arrive at the finish point in the path of D- G. The line is the shortest path to the total weight of 9185.4481. The shortest path can be found by assuming all paths are traced so that the shortest path can be found.

Scenario 3 – Find the Routes with 4 Lanes

Figure 7: Scenario 3 Dijkstra's Algorithm

In Figure 7, the circle blue is the node, red circle is the path that has not been traced and a yellow circle in the path

A - B represents the starting point while the yellow circle in the path D - H is the end point. Here is a table of results from a search shortest path contained in Scenario 3:

Table 4: Table Scenario 3

Node	Status	Weight	Route
Start	Handled	0	Start
B	Handled	3743.286	Start – B
C	Discovered	6250.2344	Start – B – C
D	Discovered	7928.2931	Start – B – – D
E	Handled	4308.0653	Start – B – F – E
F	Handled	6837.0604	Start – B – F
G	Discovered	10825.8612	Start – B – E – G
H	Handled	8672.2603	Start – B – F – E – H
Finish	Handled	8895.2734	Start – B – F – E – H – Finish

Based on Table 4, the shortest route gained from the implementation of Dijkstra's algorithm in scenario 1 is the start - B - F - E - H - finish with a total weight of 9680.0833. Total weight is much shorter than the last point found is point G with a total weight of 6837.0604 coupled with the weight towards the point H is 2999.0768 bringing the total weight to 9836.1372. In contrast to the total weight if through these start - B - F - E - H which only had a total weight of 8672.2603. To reach the destination, all the points that are available to be found to compare which path is shorter. A comparison of the path is found, the status "found" changed to "visit" when the weight is on the line is shorter than the path being compared. The Figure 8 describes the result of the shortest path taken by the Dijkstra's algorithm:

Figure 8: Result of Scenario 3. Dijkstra's Algorithm

The path taken A- B are on the path that starts from the start point, then to point B, F, E, H, and then arrive at the finish point in the path D - H. The line is the shortest path compared to other lines with a total 9680.0833 weight. The shortest path can be found by assuming all paths are traced so that the shortest path can be found. Based on the result, this application can be enabling users to search for bus stop, provide recommendations bus types then users are not confused in choosing the type of bus and provide recommendations shortest path toward the bus stop.

CONCLUSIONS

Based on the results and analysis of the research that has been done, the user can find the path provided in accordance with the intended stop using a mobile application that is already available on the Google Play Store. Dijkstra's algorithm can also be implemented in the shortest route search in mapping, using Google Maps API. A heuristic solution procedure, based on DIJKSTRA Algorithm is developed to simultaneously investigate the issues. The proposed procedures are demonstrated through an illustrative example, by using three different scenarios in mobile application. Opportunities for future research may consist in the consideration of various types of collaborations to extend of the proposed procedure.

REFERENCES

1. E. Turban, David K, J. Lee, T. Liang, D. Turban. (2012). *"Electronic Commerce"*, 7th Edition. United State: Pearson.
2. Budiarsyah, DibiKhairurrazi. *Algoritma DIJKSTRA, Bellman-Ford, dan Floyd- Warshall untuk Mencari Rute Terpendek dari Suatu Graf*, Makalah, 2010
3. Ni Kai, Zhang Yao-ting, Ma Yue-peng., "Shortest Path Analysis Based on Dijkstra's Algorithm in Emergency Response System", *TELKOMNIKA Indonesian Journal of Electrical Engineering*, vol.12, No.5, pp. 3476-3482, May 2014.
4. Cormen, Thomas H., *"Introduction to Algorithms"* Third Edition. The MIT Press Cambridge. London, 2009.
5. O'Brien, James A., *"Introduction to Information Systems"*, 12th edition. McGrawHill, 2005.
6. Edmonds, Jeff., *"How to Think About Algorithms"*. Cambridge University Press. New York, 2010.
7. Kleinberg, John., *"Algorithm Design"*. Pearson Education. Boston, 2006.
8. Kendall, K.E, J.E. Kendall., *"System Analysis and Design"*, 5th Edition, Prentice hall, 2004.