Optimizing Taxi Booking Application System By Dijkstra Algorithm

Lie, Albert Tri Adrian 13513076
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
alberttriadrian@students.itb.ac.id

***Abstract*—Djikstra algorithm is an improved concept of greedy algorithm from algorithm strategy, that is used for solving the optimization problem. The Dijkstra algorithm is used by many people to determine the shortest path.**

This paper will discuss further how Dijkstra algorithm can be applied by taxi companies such as Grab Taxi, Easy Taxi, and so on. The problem is how they can determine which taxi that should be used for going to the certain customer. In this context, Dijkstra algorithm can be used to decide which taxi that will be more optimal and faster to pick up customer so that the customer doesn't need to wait for a long time. By harnessing this algorithm, taxi companies can optimize the transportation cost and waiting time of the customer because this algorithm can find the shortest path and alternative shortest path and know the estimation time as well.

***Keywords*—Dijkstra, Greedy, Taxi, Optimization, shortest path.**

I. INTRODUCTION

Currently taxi has been growing significantly. Almost anytime and anywhere, we can find taxi and those transportation are really helpful for people to go to some places. And for making customer more comfortable, many taxi companies provide taxi booking application so that people can book the taxi immediately and conveniently. Many taxi companies such as Grab Taxi, Easy Taxi, and so on have provided those services whether it is mobile or mobile phone based.

Source : http://grabtaxi.com/jakarta-indonesia/

Figure 1.1 Grab Taxi Company

Source : http://www.easytaxi.com/id

Figure 1.2 Easy Taxi Company

It is a great opportunity for all taxi companies because as we know that the internet user in Indonesia are growing every year. This phenomenon shows that many people love doing their activities via internet including the services liked booking taxi.


Indonesia Internet Users

45,222,323

Internet Users in Indonesia

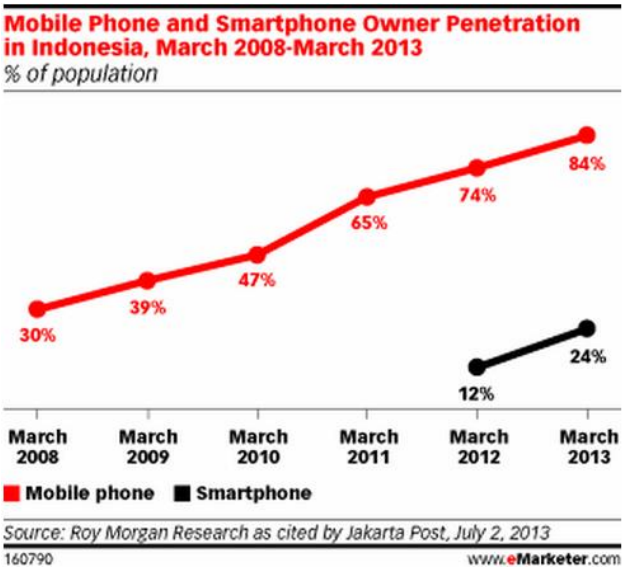
Indonesia

Year (July 1)	Internet Users**	User Growth	New Users	Country Population	Population Change	Penetration (% of Pop. with Internet)	Country's Share of World Population	Country's Share of World Internet Users	Global Rank
2014*	42,258,824	9%	3,468,057	252,812,245	1.18%	16.72%	3.49%	1.45%	12
2013*	38,790,767	2%	872,427	249,865,631	1.22%	15.52%	3.49%	1.43%	12
2012	37,918,340	27%	7,979,498	246,864,191	1.26%	15.36%	3.49%	1.51%	12

Source : http://www.internetlivestats.com/internet-users-by-country/

Figure 1.3 Indonesia Internet User

In addition, the number of people that use the mobile phone is increasing sharply. It can be positive impact for the taxi companies to expand their mobile booking app via mobile phone. As long as the number of mobile phone user grow, the number of people that use their services will also grow.



Source : http://www.emarketer.com/Article/Smartphone-Penetration-Doubles-Indonesia/1010102

Figure 1.4 Mobile Phone Penetration in Indonesia

Indeed, taxi booking app solution can be a market place between taxi driver and customer instantly and easily. However, there still exists problem whether it is from customer or the taxi companies itself.

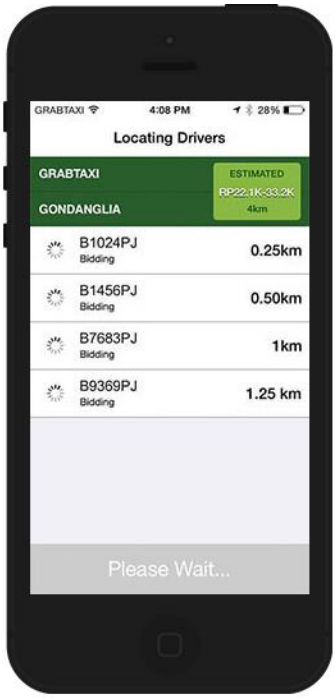
The problem that is existed from the customer is they are waiting for a long time so that it will decrease the customer rate of the taxi companies. It will cause the decreasing performance of taxi companies.

The problem that is existed from the taxi companies is they waste money much more than the expected cost. The cost in this context is the cost that is used to taxi transportation including the fuel of taxi.

Those problems will continuously occur as long as the companies cannot estimate the effective path (shortest path) for each customer's order.

Dijkstra algorithm can support to solve those problems from customer and taxi companies side. In one example of taxi booking application it est Grab Taxi, the customer can see and find the near taxi and estimation time from their place.

This thing will make companies easier on deciding and calling which taxi that should currently go to certain customer so that it will result in shortest distance and cost much less fuel for each taxi.



Source : http://grabtaxi.com/jakarta-indonesia/

Figure 1.5 Grab Taxi Interface Example

That's why Dijkstra algorithm is really helpful for solving this issue.

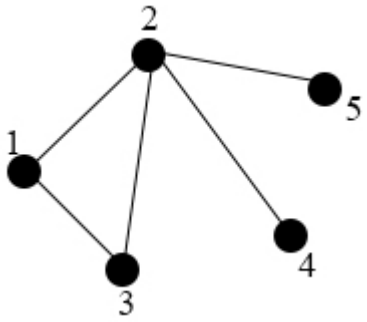
II. THEORY

II.1 Graph

Graphs are discrete structures consisting of vertices and edges that connect to these vertices. Mathematically, graph is defined as set (V, E) where [1]:

- V = a nonempty set of vertices = {v1, v2, v3, …, vn}
- E = a set of edges that connect a pair of vertices. Each edge has either one or two vertices associated with it, called endpoints. An edge is said to connect its endpoints. E = {e1, e2, e3, …, en}.

Graph can be grouped into several categories regarding whether edges have direction, same pair of vertices, or loops are allowed. Here is the example of graph:



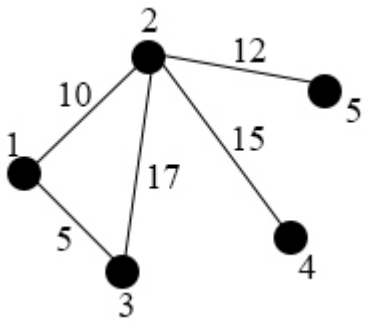
Source :
http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2013-2014/Makalah2013/MakalahIF2120-2013-016.pdf

Figure 2.1 A Graph with 5 Vertices and 5 Edges

The graph above is undirected graph with 5 vertices and 5 edges. The vertices are 1, 2, 3, 4, 5.

II.2 Weighted Graph

A weighted graph is a graph G in which each edge e has been assigned a real number called weight of e [1] This weight of each edge represents the distance between two vertices. Below is the example of weighted graph.



Source :
http://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2013-2014/Makalah2013/MakalahIF2120-2013-016.pdf

Figure 2.2 Weighted Graph

The weighted graph above means that every edge between to vertices has weight that represents the distance between them.

II.3 Greedy Algorithm

Greedy algorithm is algorithm used for deciding the result from optimization problem. There are two optimization problems it est maximum and minimum. On each movement, decision that is achieved is a local optimal value that is got from global optimal value, hoping that the result is not too far with the real global optimal value.

There are two requirements of greedy algorithm :

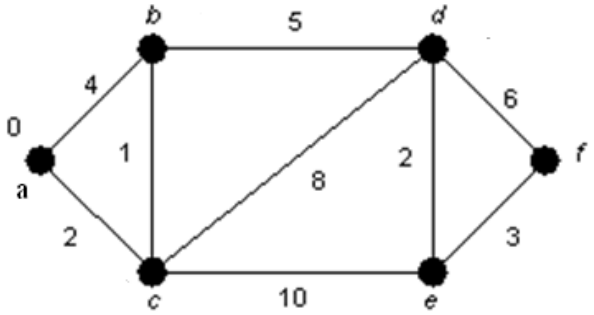
a. *Greedy Choice Property*:
   We can achieve the global optimal from local optimal without changing or considering the previous decision.
b. *Optimal Substructure Property*:
   We can achieve the optimal solution by considering the sub solution of the problem.

The elements of greedy algorithm :

a. Candidate Set, C
b. Solution Set, S
c. Selection Function
d. Feasible Function
e. Objective Function

II.4 The Shortest Path Algorithm

The shortest path algorithm is algorithm used for determine the path from certain point to other points.



Source :
http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2013-2014-genap/Makalah2014/MakalahIF2211-2014-033.pdf

Figure 2.3 Shortest Path Graph Problem

There are 4 kinds of shortest path algorithm:

a. A pair shortest path
b. All pairs shortest path
c. Single source shortest path
d. Intermediate shortest path

II.5 Dijkstra Algorihtm

Dijkstra algorithm is the development from the greedy concept on algorithm strategy that is used as support tool in determining the shortest path in graph G = (V, E) to the remaining vertices in the graph. When the shortest path to a particular vertex is desired, the algorithm is terminated when the shortest path to that vertex has been found.

Dijsktra algorithm can solve the shortest path problem from an initial point to a destination point in a weighted graph. The shortest path is achieved from two points if total weight of all nodes is the smallest.

Before proceeding, there is some important notations that will be used in the further explanation [2]:

1. l(i , j) : distance between node i to node j.
2. a : starting point
3. d_{ai} : permanent shortest path from initial point to node i in a weighted graph.
4. q_i : last point that is considered shortest from starting point to node I in a weighted graph.
5. c : last point that has moved to permanent condition.

The following is the procedure of Dijkstra algorithm:

1. Process starts from starting point (a). Thus, the permanent point is only a. And the other points temporarily is filled with infinite value (∞).
2. Search all the branch from the last node that is permanent with this equation :

   $d_{ai} = \min[d_{ai}, d_{ac}\text{-}l(c,i)]$

3. Decide which point that will move from temporary to permanent by comparing the value of point that is acquired from (2) and take the minimum value. Then, for knowing the value of permanent point before, use it in equation:

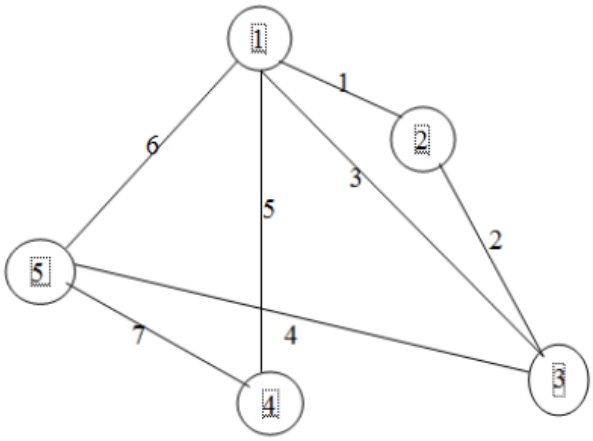
   $[d_{ai}\text{-}l(c,i)] = \ldots d_{ai}$

4. After getting the point with the smallest value, we can decide that the point is the permanent minimum value
5. If there are points that are not permanent, repeat the process from number (2).

The following is the pseudo-code of Dijkstra algorithm [2]:

```
dist[s] ← 0
for all v € V-{s}
        do dist[v] ← ∞
S ← ø
Q ← V
While Q ≠ ø
do u ← mindistance(Q,dist)
        S ← S U {u}
        for all v € neighbors[u]
                do if dist[v]>dist[u]+w(u,v)
                        then d[v] ← d[u]+w(u,v)
return dist
```

Below is the example of undirected graph that consists of five nodes and seven edges. Dijkstra algorithm can be used to find the shortest path from this weighted graph.



Source : http://eprints.dinus.ac.id/5180/1/P6-TI24-SEMANTIK-80_Farida_Ardiani-UII_Yogjakarta.pdf

Figure 2.2 Weighted Graph Example

According to the example, starting point of the graph is node 1 and the destination point is node 4.

Table 1.1 Path Table

Path	Initial Path				
	1	**2**	**3**	**4**	**5**
	0	0	0	0	0
1	1	0	0	0	0
1-2	1	1	0	0	0
2-3	0	1	1	0	0
3-5	0	0	1	0	1
5-4	0	0	0	1	1

Table 2.2 Node Table

Node	l(i,j)				
	1	**2**	**3**	**4**	**5**
1	∞	∞	∞	∞	∞
2	1	∞	∞	∞	∞
3	3	2	∞	∞	∞
4	5	∞	∞	∞	7
5	6	∞	4	∞	∞

The solution with Dijkstra algorithm from node 1 to destination point node 4 has been shown liked the table above according to the calculation of the Dijkstra algorithm. On first row, all the successors are set to 0, which means to give value for the source route that will be route and the limitation of other points, that declare the fact of unknown path.

Then, because the node one is the source or starting point, node 1 is absolutely chosen. Thus, status 0 is changed to 1. Node one will check the adjacent nodes such as node 2,3,4, and 5. From that point, Dijkstra will choose the minimal weight to go towards next node. Node 2 is chosen with weight 1, then set the status from 0 to 1 and so on. Thus, from the searching of the shortest path, we will get the minimal path according to Dijkstra calculation from node 1 to node 4 with weight 5.

The advantages of Dijkstra algorithm [4]:

a. Finds shortest path in O(E+ V Log(V)) if you use a min priority queue.

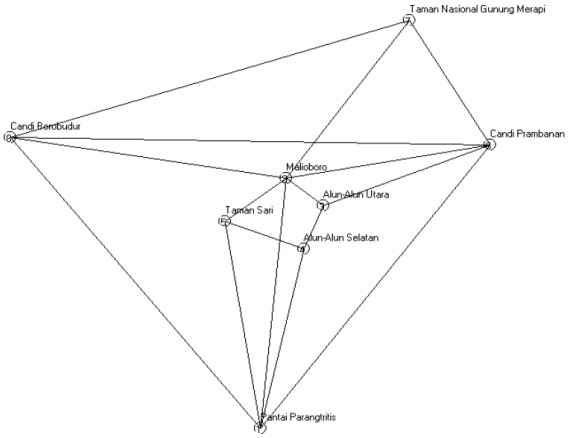
This is true only if you implement priority queue with Fibonacci heap, then amortized operation over it will take O(1). Otherwise, if you use any other implementation of priority_queue (like standard C++ STL) it should take E log(E) + V.

The disadvantages of Dijkstra algorithm [5]:

a. The major disadvantage of the algorithm is the fact that it does a blind search there by consuming a lot of time waste of necessary resources.
b. It cannot handle negative edges. This leads to acyclic graphs and most often cannot obtain the right shortest path. Decision tree can sometimes create biased tree if some classes dominate. So, it will be important to use data that are quite balanced.

III. APPLYING DIJKSTRA ALGORITHM FOR FINDING THE NEAREST TAXI

Most of the taxi booking app harnessed the map application liked Google Map but they only knew the coordinate of each places. By applying Dijkstra algorithm, they can make wise decision that will make the cost of transportation minimum or the distance is minimal regardless anywhere and anytime the customer want to book taxi. Let's use the example below:



Source : http://eprints.dinus.ac.id/5180/1/P6-TI24-SEMANTIK-80_Farida_Ardiani-UII_Yogjakarta.pdf

Figure 2.2 Problem Example

Table 3.1 Location

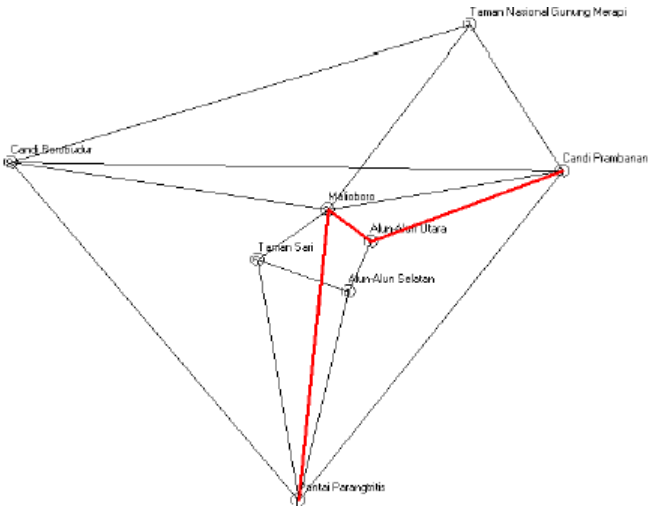
Node	Place
1	Pantai Parangtritis
2	Malioboro
3	Alun – alun Utara
4	Alun – alun Selatan
5	Taman Sari
6	Candi Prambanan
7	Taman Nasional Gunung Merapi
8	Candi Borobudur

Table 3.2 Location Distance

Path	Distance (Kilometers)
1-2	30
1-4	25
1-5	28
1-6	48
1-8	65
2-3	1
2-5	3
2-6	17
2-7	29
2-8	42
3-4	1
3-6	15
4-5	3
6-7	27
6-8	57
7-8	46

Let's say that customer suddenly want to book taxi to go towards Candi Prambanan (node 6) from Pantai Parangtritis (node 1). By applying Dijkstra algorithm process that has been explained in the previous chapter, the result will be:

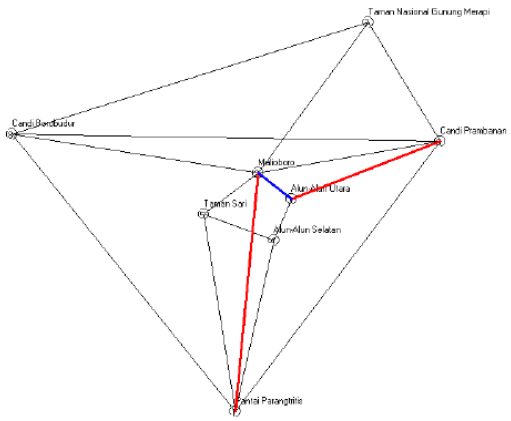
Starting node = 1; Destination point = 6
The shortest path = 48.00 km.
The speed = 60 km/hour.
The distance time = 0 hours, 46 minutes.



Source : http://eprints.dinus.ac.id/5180/1/P6-TI24-SEMANTIK-80_Farida_Ardiani-UII_Yogjakarta.pdf

Figure 2.2 Problem Example with Solution (Red line)

However, in the real life. The condition is rarely liked those example. Let's say that suddenly there are some path blocking. (the blue line is the blocking path)

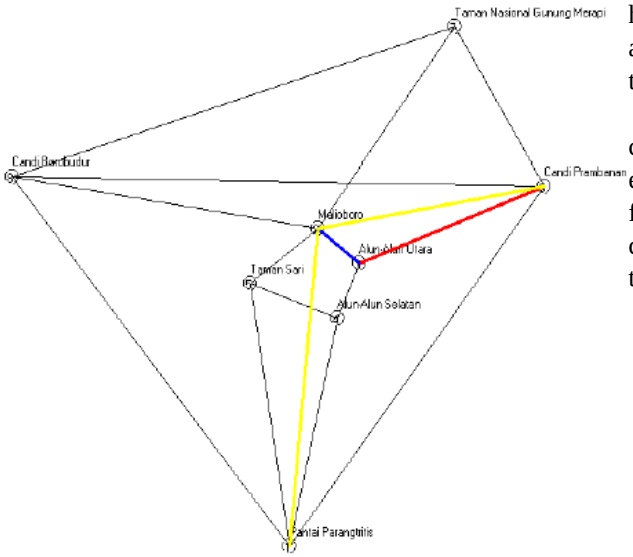


Source : http://eprints.dinus.ac.id/5180/1/P6-TI24-SEMANTIK-80_Farida_Ardiani-UII_Yogjakarta.pdf

Figure 2.2 Problem Example With Blocking Path (Blue line)

In this condition, we need to find the alternative shortest path. So, the taxi company will not lose a customer and still can do their transportation effectively. By using the Dijkstra algorithm, we can find the alternative solution although there are some constraint. And the result is:

Starting node = 1; Destination point = 6
The shortest path = 57.00 km.
The speed = 60 km/hour.
There are some blocking on the line 6.
The distance time = 0 hours, 47 minutes.



Source : http://eprints.dinus.ac.id/5180/1/P6-TI24-SEMANTIK-80_Farida_Ardiani-UII_Yogjakarta.pdf

Figure 2.2 Problem Example With Alternative Solution (Yellow line)

From the results above, we understand that the Dijkstra algorithm is really effective for estimating the shortest path to the certain place. Many application nowadays have used this algorithm to determine their shortest route.

The basic idea behind the Dijkstra algorithm is to get the shortest path. Because our assumption is the distance in the real world never be negative, we can be tolerant with the drawback of using Dijkstra algorithm. That way, we can get to the correct decision of the distance of taxi to customer.

The main drawback notices from this study is that if there is many blocking way, maybe the decision of the accuracy of Dijkstra algorithm will be decreased.

IV. CONCLUSION

In this paper, we implement Dikstra algorithm, the improvement of greedy algorithm from algorithm strategy to finding the nearest taxi in taxi booking application of several taxi companies. This paper mainly focuses on one shortest path approach. It is shortest path problem with Dijkstra algorithm. The analysis of this problem using data of location of real places in Indonesia with Dijkstra approach calculation to find the shortest path and alternative shortest path.

Dijkstra algorithm is powerful because in the real life there are no negative distance so that Dijkstra algorithm can prevent its drawback and use its advantages to find the optimal solution. However, if there are such a thing like blocking path or something like that, it s=will decrease the accuracy of Dijkstra algorithm. So it will be really helpful if we assume the condition is normal, because many application now has used this algorithm and have been proven to be accurate. It would be good approach for taxi companies to decrease the cost of transportation and waiting time for its customer.

In future, more research to overcome the blocking path case in Dijkstra algorithm is required to apply the more effective approach for finding the shortest path, not only for taxi but also for all public transportation. Thus, the crowded condition of Indonesia can will be decreased and the economic cost of transportation can be redeemed.

V. APPENDIX

a) Node: the part of the tree that contain some information or value.
b) Edge: the connector of two nodes

VII. ACKNOWLEDGMENT

Lie, Albert Tri Adrian as the author of this paper, want to express his deepest gratitude to Dr. Ir. Rinaldi Munir, M.T. and Drs Nur Ulfa as the lecturers of IF 2211 – “Strategi Algoritma”. Special thanks to all of my family, my friends, and other people that give any form of support to me to finish this paper.

REFERENCES

[1] Munir, Rinaldi. Matematika Diskrit Edisi Kedua, Bandung : Informatika, 2001, hlm. 291, 294.

[2] Ratnasari, Asti."Penentuan Jarak Terpendek dan Jarak Terpendek Alternatif Menggunakan Algoritma Dijkstra serta Estimasi Waktu Tempuh", Yogyakarta : Universitas Islam Indonesia. 2013.

[3] R. A. D. Novandi, "Perbandingan Algoritma Dijkstra dan Algoritma Floyd-Warshall dalam Penentuan Lintasan (*Single Pair Shortest Path*)", Bandung : Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung. 2007.

[4] http://www.quora.com/What-are-advantages-and-disadvantages-of-Dijkstras-algorithm-in-data-structures . Access : 3rd May 2015.

[5] http://www.let.rug.nl/~vannoord/papers/nle/node35.html Acess : 3rd May 2015.

[6] http://www.cs.nyu.edu/courses/summer07/G22.2340-001/Presentations/Puthuparampil.pdf. Access 3rd May 2015.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 5 Mei 2015

Lie, Albert Tri Adrian
13513076