RESEARCH ARTICLE OPEN ACCESS

Optimization of VSAT IP Installation Routes using the Dijkstra Algorithm

Ravi Suherman[1], Nia Rahma Kurnianda[2]
[1]Computer Science Faculty, Mercu Buana University, Jakarta
Email: 41816120118@student.mercubuana.ac.id
[2]Computer Science Faculty, Mercu Buana University, Jakarta
Email: nia.rahma@mercubuana.ac.id

****

Abstract:

The development of technology is increasingly fast—one of the technology companies in Indonesia, namely PT. Telkom Indonesia has a subsidiary, namely PT. Telkom Satelit Indonesia (Telkomsat) where PT. Telkom Satelit Indonesia (Telkomsat) is a satellite-based internet service provider company. One of the products of PT. Telkom Satelit Indonesia (Telkomsat), namely VSAT IP. However, there are often problems in the field, namely the delay in installation time. One of the factors causing this problem is the inaccurate selection of the closest location point installed first. Therefore, the purpose of this study is to provide advice on which location points to take precedence based on the nearest distance. One of the algorithms that can accommodate this is the Dijkstra algorithm. This research shows that Dijkstra's Algorithm can provide the products of which location points have the closest distance.

Keywords — **Dijkstra's Algorithm, Route Optimalize, Shortest Route**

****

I. INTRODUCTION

In the current era, technological developments are increasingly showing a high level of development. Almost all aspects of human life have been touched by technology. One of the existing technological developments is the use of satellites as internet service providers.

PT. Telkom Satelit Indonesia (Telkomsat) is a subsidiary company of PT. Telkom Indonesia, which is engaged in providing satellite services [1]. VSAT IP is one of the service products offered by PT. Telkom Satelit Indonesia.

However, during the new VSAT IP installation process, problems were often encountered, one of which was a delay or exceeding the predetermined period, so the service provider had to pay several compensation fees.

This is undoubtedly a serious problem. If allowed to continue, it will indeed affect the operational costs and company performance. Therefore, we need a solution to minimize these losses. One solution that can be offered is to know which location points have the closest distance to the PT. Telkom Satelit Indonesia (Telkomsat).

Dijkstra's algorithm is one of the algorithms that can help optimize the shortest route from a location. By implementing Dijkstra's algorithm, it is hoped that it can assist in making decisions on the closest new tide location.

II. LITERATURE STUDY

This study, using several theories related to the research topic, including:

A. Dijkstra's Algorithm

A Dijkstra's algorithm was first introduced by Edsger W. Dijkstra [2]. Dijkstra's algorithm is an algorithm that can be used to determine the shortest path in a weighted graph without being convoluted [3]. The stages in the Dijkstra algorithm include [4]:

1. Specify the departure point or node and set the weight to 0
2. Assigning weight to each point or node
3. Set all points or nodes that have not been passed as departure points or nodes
4. From the departure point or node, consider the untreated point or node and calculate the departure point or node's distance. If there is a point or node that has a smaller distance, the previous node will be deleted
5. Put a period or node that has been traversed before. Points or nodes that have been traveled back will not be considered or rechecked, and the stored distance is the distance that has the most negligible value
6. If there is a point or node that has never been traversed, make it the next point or node of departure

B. Graph

A mathematician from Switzerland, Leonhard Euler 1735, introduced graph theory to solve Konigsberg Bridge's problem[5]. The graph is a collection of several points (nodes) connected by lines (edge) [6]. The graph is a pair of Vertex (V) and Edge (E) set. Vertex itself is a point or node, while the edge is a line [7].

C. Spanning Tree

The tree is an undirected graph, and no loop is connected and acyclic[3]. A tree can be a spanning tree on a graph if the tree subgraph of the graph and the tree contains all the points or nodes of the graph [8].

D. Shortest Route

The graph used in determining the shortest path is a weighted graph. If on a track M and V0 are the origin or departure points, and Vn is the destination point, the shortest path notation is Pt, where Pt is the t-the path from V0 to Vn[9].

$$l_t(V0, Vn) = \min_{i=1,\ldots,m} l_i\,(V0, Vn)$$

E. Pyhton

In 1989, Guido van Rossum developed Python and in 1991 introduced Python[10][11]. Python is a high-level programming language that is a development of the ABC programming language[12].

III. RESEARH METHOD

A. Research Location

The research location is in the office of PT. Telkom Satelit Indonesia (Telkomsat) Cibubur which is located at Jl. Jl. Pringgondani 2 No.10-33, Harjamukti, Kec. Cimanggis, City of Depok, West Java 16454 [1].

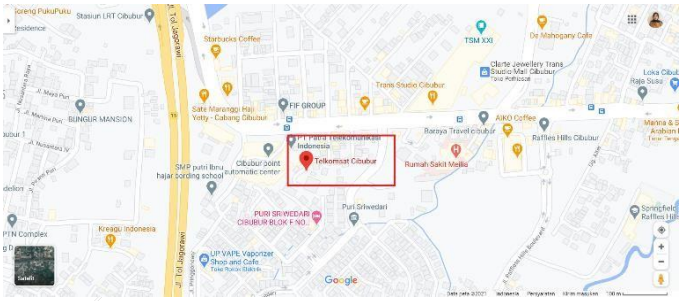

Fig. 1 Research Location

B. Data Collection Method

The data collection technique used is to use literature studies by looking for books or journals related to the research topic.

IV. RESULT AND DISCUSSION

A. Dataset

1. Cimanggis district

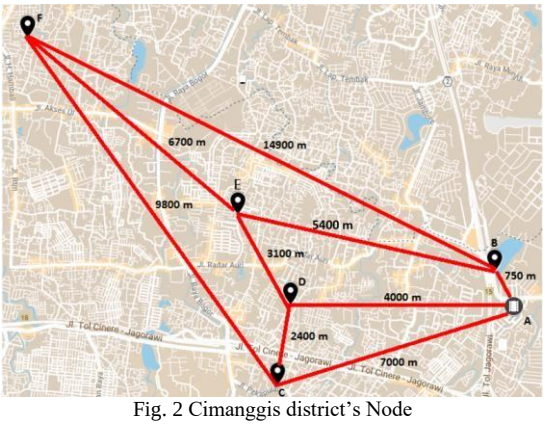

Fig. 2 Cimanggis district's Node

Node A : Kantor Telkomsat Office

Node B : Harjamukti village
Node C : Curug village
Node D : Cisalak pasar village
Node E : Mekarsari village
Node F : Pasir Gunung Selatan village

2. Beji district

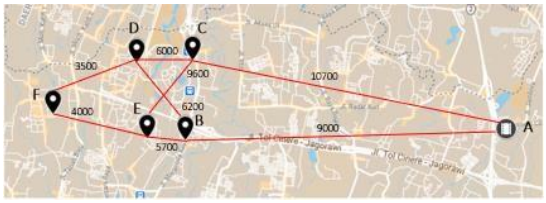

Fig. 3 Beji district's Node

Node A : Kantor Telkomsat Office
Node B : Kemiri Muka village
Node C : Pondok Cina village
Node D : Kukusan village
Node E : Beji Timur village
Node F : Tanah Baru village

3. Bojongsari district

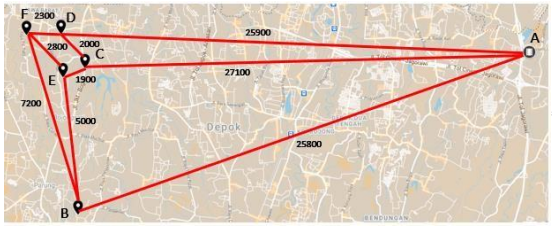

Fig. 4 Bojongsari district's Node

Node A : Telkomsat Cibubur Office
Node B : Duren Seribu village
Node C : Bojongsari Baru village
Node D : Serua village
Node E : Bojongsari Lama village
Node F : Pondok Petir village

B. Dijkstra's Algorithm Implementation

To get a comparison, the results processed using Python will be compared with manual calculations.

1. Cimanggis district
   Manual calculation results:

Vertic	A	B	C	D	E	F
A	0 A	750 A	7000 A	4000 A	∞	∞
B		750 A	7000 A	4000 A	6150 B	15650 B
D			6400 D	4000 A	6150 B	15650 B
E			6400 D		6150 B	12850 E
C			6400 D			12850 E
F						12850 E

Fig. 5 Manual Calculation Result in Kec. Cimanggis

The results of calculations using Python:

```
graph = {'Telkomsat':{'Harjamukti':750,'Curug':7000,'Cisalak Pasar':4000},
         'Harjamukti':{'Mekarsari':5400,'Pasir Gunung Selatan':14900},
         'Curug':{'Cisalak Pasar':2400,'Pasir Gunung Selatan':9800},
         'Cisalak Pasar':{'Mekarsari':3100},
         'Mekarsari':{'Pasir Gunung Selatan':6700},
         'Pasir Gunung Selatan':{}}

def dijkstra(graph,start,goal):
    shortest_distance = {}
    predecessor = {}
    unseenNodes = graph
    infinity = 9999999999999
    path = []
    for node in unseenNodes:
        shortest_distance[node] = infinity
    shortest_distance[start] = 0

    while unseenNodes:
        minNode = None
        for node in unseenNodes:
            if minNode is None:
                minNode = node
            elif shortest_distance[node] < shortest_distance[minNode]:
                minNode = node

        for childNode, weight in graph[minNode].items():
            if weight + shortest_distance[minNode] < shortest_distance[childNode]:
                shortest_distance[childNode] = weight + shortest_distance[minNode]
                predecessor[childNode] = minNode
        unseenNodes.pop(minNode)

    currentNode = goal
    while currentNode != start:
        try:
            path.insert(0,currentNode)
            currentNode = predecessor[currentNode]
        except KeyError:
            print('Path not reachable')
            break
    path.insert(0,start)
    if shortest_distance[goal] != infinity:
        print('Jarak terpendek yang ditempuh ' + str(shortest_distance[goal]))
        print('Dimana titik yang akan dilalui yaitu ' + str(path))

dijkstra(graph, 'Telkomsat', 'Pasir Gunung Selatan')
```

```
Jarak terpendek yang ditempuh 12850
Dimana titik yang akan dilalui yaitu ['Telkomsat', 'Harjamukti', 'Mekarsari', 'Pasir Gunung Selatan']
```

Fig. 6 The results of calculations using Python in Kec. Cimanggis

The calculations manually and using Python, the shortest route generated, are the location point A-B-E-F or the Telkomsat Cibubur Office, Harjamukti village, Mekarsari village and Pasir Gunung Selatan village.

2. Beji district
   Manual calculation results:

Vertic	A	B	C	D	E	F
A	0 A	9000 A	10700 A	∞	∞	∞
B		9000 A	10700 A	15200B	14700 B	∞
C			10700 A	15200B	14700 B	∞
E				15200B	14700 B	18700 E
F				15200B		18700 E
D						18700 E

Fig. 7 Manual Calculation Result in Kec. Beji

The results of calculations using Python:

```
graph = {'Telkomsat':{'Kemiri Muka':9000,'Pondok Cina':18700},
         'Kemiri Muka':{'Kukusan':6200,'Beji Timur':5700},
         'Pondok Cina':{'Kukusan':6000,'Beji Timur':9600},
         'Kukusan':{'Tanah Baru':3500},
         'Beji Timur':{'Tanah Baru':4000},
         'Tanah Baru':{}}

def dijkstra(graph,start,goal):
    shortest_distance = {}
    predecessor = {}
    unseenNodes = graph
    infinity = 9999999999999
    path = []
    for node in unseenNodes:
        shortest_distance[node] = infinity
    shortest_distance[start] = 0

    while unseenNodes:
        minNode = None
        for node in unseenNodes:
            if minNode is None:
                minNode = node
            elif shortest_distance[node] < shortest_distance[minNode]:
                minNode = node

        for childNode, weight in graph[minNode].items():
            if weight + shortest_distance[minNode] < shortest_distance[childNode]:
                shortest_distance[childNode] = weight + shortest_distance[minNode]
                predecessor[childNode] = minNode
        unseenNodes.pop(minNode)

    currentNode = goal
    while currentNode != start:
        try:
            path.insert(0,currentNode)
            currentNode = predecessor[currentNode]
        except KeyError:
            print('Path not reachable')
            break
    path.insert(0,start)
    if shortest_distance[goal] != infinity:
        print('Jarak terpendek yang ditempuh ' + str(shortest_distance[goal]))
        print('Dimana titik yang akan dilalui yaitu ' + str(path))


dijkstra(graph, 'Telkomsat', 'Tanah Baru')
```

```
Jarak terpendek yang ditempuh 18700
Dimana titik yang akan dilalui yaitu ['Telkomsat', 'Kemiri Muka', 'Beji Timur', 'Tanah Baru']
```

Fig. 8 The results of calculations using Python in Kec. Beji

The calculations manually and using Python, the shortest route generated, are the location point A-B-E-F or the Telkomsat Cibubur Office, Kemiri Muka village, Beji Timur village and Tanah Baru village.

3. Bojongsari district

Manual calculation results:

Vertic	A	B	C	D	E	F
A	0 A	25800 A	27100 A	25900 A	∞	∞
B		25800 A	27100 A	25900 A	30.800 B	33000 B
D			27100 A	25900 A	30.800 B	28200 D
C			27100 A		29000 C	28200 D
F						28200 D
E						

Fig. 9 Manual Calculation Result in Kec. Bojongsari

The results of calculations using Python:

```
graph = {'Telkomsat':{'Duren Seribu':25800,'Bojongsari Baru':27100,'Serua':25900},
         'Duren Seribu':{'Bojongsari Lama':5000,'Pondok Petir':7200},
         'Bojongsari Baru':{'Serua':25900,'Bojongsari Lama':5000},
         'Serua':{'Pondok Petir':7200},
         'Bojongsari Lama':{'Pondok Petir':7200},
         'Pondok Petir':{}}

def dijkstra(graph,start,goal):
    shortest_distance = {}
    predecessor = {}
    unseenNodes = graph
    infinity = 9999999999999
    path = []
    for node in unseenNodes:
        shortest_distance[node] = infinity
    shortest_distance[start] = 0

    while unseenNodes:
        minNode = None
        for node in unseenNodes:
            if minNode is None:
                minNode = node
            elif shortest_distance[node] < shortest_distance[minNode]:
                minNode = node

        for childNode, weight in graph[minNode].items():
            if weight + shortest_distance[minNode] < shortest_distance[childNode]:
                shortest_distance[childNode] = weight + shortest_distance[minNode]
                predecessor[childNode] = minNode
        unseenNodes.pop(minNode)

    currentNode = goal
    while currentNode != start:
        try:
            path.insert(0,currentNode)
            currentNode = predecessor[currentNode]
        except KeyError:
            print('Path not reachable')
            break
    path.insert(0,start)
    if shortest_distance[goal] != infinity:
        print('Jarak terpendek yang ditempuh ' + str(shortest_distance[goal]))
        print('Dimana titik yang akan dilalui yaitu ' + str(path))


dijkstra(graph, 'Telkomsat', 'Pondok Petir')
```

```
Jarak terpendek yang ditempuh 28200
Dimana titik yang akan dilalui yaitu ['Telkomsat', 'Serua', 'Pondok Petir']
```

Fig. 10 The results of calculations using Python in Kec. Bojongsari

The calculations manually and using Python, the shortest route generated, are the location point A-D-F or the Telkomsat Cibubur Office, Serua village and Pondok Petir village.

C. Evaluation

To find out whether the results obtained are accurate or not, an evaluation process is required. The following are the results of the evaluation obtained:

Optimization of VSAT IP Installation Routes using the Dijkstra Algorithm					
#	Installation Location	Reference Value (RV)	Measurement Value (MV)	Error (RV-MV)	Error % (Error/RVx 100%
1	Cimanggis Village	12850	12850	0	0
2	Beji Village	18700	18700	0	0
2	Bojongsari Village	28200	28200	0	0
				Total error %	0%
				Average error %	0%

Fig. 11 Evaluation Results

#	Accuration Results Range	Scale	Description
1	0%-20%	1	Very bad
2	21%-40%	2	Bad
3	41%-60%	3	Sufficient
4	61%-80%	4	Good
5	81%-100%	5	Very good

Fig. 12 Accuration Results Classification

Based on the evaluation results in Fig. 11 and the result clacification table in Fig. 12, shows that the accuracy results show very good category, so it is suitable to be applied in the optimization of the VSAT IP installation route at PT. Telkom Satelit Indonesia (Telkomsat)..

V. CONCLUSION

Dijkstra's algorithm proved that it can be used to see potential shortest route for VSAT IP installation in Cimanggis district, Beji district, and Bojongsari district. In Cimanggis district, the location points that have the shortest distance are the Cibubur Telkomsat Office, Harjamukti village, Mekarsari village, and Pasir Gunung Selatan village, with a total distance of 12850 m. In Beji district, the location points that have the shortest distance are the Cibubur Telkomsat Office, Kemiri Muka village, Beji Timur village, and Tanah Baru village, with a total distance of 18700 m. In Bojongsari district, the location points that have the shortest distance are the Cibubur Telkomsat Office, Serua village, and Pondok Petir village, the total distance is 28200 m. And the accuracy results show that the calculation results fall into the very good category

REFERENCES

[1] "Website Telkomsat." https://www.telkomsat.co.id/id/produk/vsat-ip.

[2] F. Daniel and P. N. . Taneo, *Teori Graf*. Yogyakarta: Penerbit Deepublish, 2019.

[3] Marsudi, *Teori Graf*. Malang: UB Press, 2016.

[4] R. Dwi, Saputra and Ardana, "Penerapan Algoritma Dijkstra pada Aplikasi Pencarian Rute Bus Trans Semarang," *Skripsi Jur. Ilmu Komputer, Fak. Sains Dan Mat. Univ. Diponegoro*, no. Snik, pp. 299–306, 2016.

[5] A. P. Rahadi, "Penjadwalan menggunakan Pewarnaan Graf dengan Algoritma Largest First," *J. Padegogik*, vol. 2, pp. 1–13, 2019.

[6] A. R. Kadafi, E. S. Budi, R. Bagus, and D. Putra, "Pemanfaatan Pohon Keputusan dan Graf Dalam Pemetaan Karyawan Berbasis Kompetensi," vol. 6, no. 6, pp. 563–569, 2019.

[7] N. Yannuansa and R. Ramadhani, "PENGGUNAAN GRAF DALAM PENJADWALAN PERKULIAHAN TEKNIK ELEKTRO UNIVERSITAS HASYIM ASYARI.pdf." 2018.

[8] I. Rina, D. Sulistiowati, and I. D. Rianjaya, "MENENTUKAN MINIMUM SPANNING TREE PADA JARINGAN PLN DI DETERMINING MINIMUM SPANNING TREE IN PLN NETWORK IN PADANG SUB-DISTRICT," 2019.

[9] A. Zaki, "ALGORITMA DIJKSTRA: TEORI DAN APLIKASINYA," vol. VI, no. 4, pp. 1–8.

[10] Y. Supardi and Y. Syarief, *Tip dan Trik Program Database Python*. Jakarta: Elex Media Komputindo, 2020.

[11] S. A. Qutsiah, M. K. Sophan, and Y. F. Hendrawan, "PEMBERLAJARAN MATEMATIKA DATAR MENGGUNAKAN PYTHON PADA PERANGKAT," vol. XI, 2016.

[12] S. H. . Herho, "Tutorial Pemrograman Python 2 Untuk Pemula."