Perbandingan Algoritma Astar dan Dijkistra Dalam Menentukan Rute Terdekat

Astar and Dijkistra Algorithm Comparison for Determining the Shortest Route

Arif Cahyo Prasetyo [1], **Maful Prayoga Arnandi** [2], **Harish Setyo Hudnanto**[3], **Bayu Setiaji**[4]
[1)2)3)4)] Fakultas Ilmu Komputer, Universitas AMIKOM Yogyakarta
Jl.Ring Road Utara, Condong Catur, Depok, Sleman, Yogyakarta 55281
Email : arif.0009@students.amikom.ac.id[2)], maful.arnandi@students.amikom.ac.id[3)],
harish.0048@students.amikom.ac.id[4)] , bayusetiaji@amikom.ac.id[4)]

Abstrak

Google Maps merupakan peta digital yang telah sering digunakan oleh masyarakat. Dengan adanya Kecerdasan Buatan dan Algoritma Pencarian rute terdekat pada Google Maps, kini semakin dimudahkan untuk mencari lokasi dari rute yang terdekat sehingga lebih efisien dalam hal waktu dan tenaga. Dalam implementasinya, algoritma pencarian sangat berguna dalam mencari rute terdekat. Diantaranya, Algoritma A (A Star) dan Algoritma Dijkstra. Kedua Algoritma tersebut bekerja dengan Mekanisme yang berbeda. Perbedaan tersebut dibandingkan dengan menggunakan Pathfinding.js. Output yang dikeluarkan diambil berdasarkan lamanya waktu pemrosesan dari setiap algoritma dalam menentukan jarak terdekat. Semakin cepat waktu pemrosesan suatu Algoritma, maka akan semakin baik untuk diimplementasikan ke dalam suatu aplikasi. Setelah melakukan penelitian didapatkan hasil bahwa kinerja Algoritma A* lebih baik dari Algoritma Dijkstra dengan rata-rata waktu 0.37 ms dan jumlah langkah sebanyak 200, sedangkan algoritma Dijkstra mempunyai rata-rata waktu 0.41 ms dan jumlah langkah 497. Untuk penelitian selanjutnya diharapkan mampu membandingkan kedua algoritma tersebut dengan parameter yang lain sehingga didapatkan rute terdekat dengan waktu tercepat.*
Kata kunci*— Google Maps, Algoritma A*, Algoritma Dijkstra*

Abstract

*Google Maps is a digital map that is often used by the public. With Artificial Intelligence and the Search Algorithm for the closest route on Google Maps, it is easier to finding the nearest location and route so make it more efficient in terms of time and effort. In its implementation, the search algorithm is very useful in finding the closest route. Among them, there are Algorithms A * (A Star) and Dijkstra Algorithms. Both of these algorithms work with different mechanisms. Its different compared with a tool named PathFinding.js. The output is taken based on the length of processing time of each algorithm in determining the closest distance. After doing the research, it was found that the performance of the A * Algorithm is better than the Dijkstra Algorithm with an average time of 0.733 ms and the number of steps is 200, while the Dijkstra algorithm has an average time of 0.933 ms and the number of steps is 497. For further research, it is expected to be able to compare the two algorithms with other parameter to get the shortest route and fastest time.*
Keywords*— Google Maps, A* Algorithm, Dijkstra Algorithm*

1. PENDAHULUAN

Di era globalisasi, penggunaan peta atau maps sudah tidak menggunakan kertas maupun buku Atlas. Kini, maps sudah terdigitalisasi untuk mempermudah penggunaan. Karenanya, diluncurkan Google Maps sebagai alat penunjuk arah dan tempat yang mampu mencakup hampir seluruh daerah di dunia. Dalam penggunaannya, Google Maps menggunakan algoritma untuk menentukan rute terpendek agar pengguna dapat tiba di lokasi tujuan dengan waktu yang lebih singkat.[1]

Algoritma untuk menentukan rute terpendek ada bermacam-macam, diantaranya adalah Algoritma A* (A star) yang melakukan pencarian dengan heuristik dan Algoritma Djikstra yang melakukan pencarian dengan blind method atau pencarian buta. Algoritma A* adalah salah satu algoritma pencarian yang menganalisa input, mengevaluasi sejumlah jalur yang mungkin dilewati dan menghasilkan solusi. Algoritma A* adalah algoritma komputer yang digunakan secara luas dalam graph traversal dan penemuan jalur serta proses perencanaan jalur yang bisa dilewati secara efisien di sekitar titik-titik yang disebut node. Algoritma Dijkstra dinamai sesuai dengan nama penemunya yaitu Edsger Dijkstra. Algoritma Dijkstra menggunakan prinsip greedy, dimana pada setiap langkah dipilih sisi dengan bobot minimum yang menghubungkan sebuah simpul yang sudah terpilih dengan simpul lain yang belum terpilih [3].

Banyak penelitian yang sudah melakukan hal serupa. Diantaranya penggunaan algoritma yang sama untuk mencari rute terdekat di tempat-tempat wisata dan penerapan dalam *game*. Di penelitian ini terdapat beberapa perbedaan dengan penelitian yang sudah dilakukan sebelumnya. Antara lain, menggunakan objek yang real yaitu gambaran dari Google Maps dengan lokasi yang sebenarnya. Selain itu, penelitian ini menggunakan alat yang berbeda yaitu PathFinding.js dan parameter yang digunakan juga berbeda. Dengan dasar tersebut, akan dibandingkan kinerja dari Algoritma A* dan Algoritma Dijkstra dalam menentukan waktu proses pencarian rute terdekat dengan object Google Maps. Dibawah ini merupakan tabel penelitian sebelumnya yang memiliki kesamaan topik serta perbedaanya dengan penelitian yang kami lakukan.

Tabel 1. Perbandingan Penelitian Sebelumnya

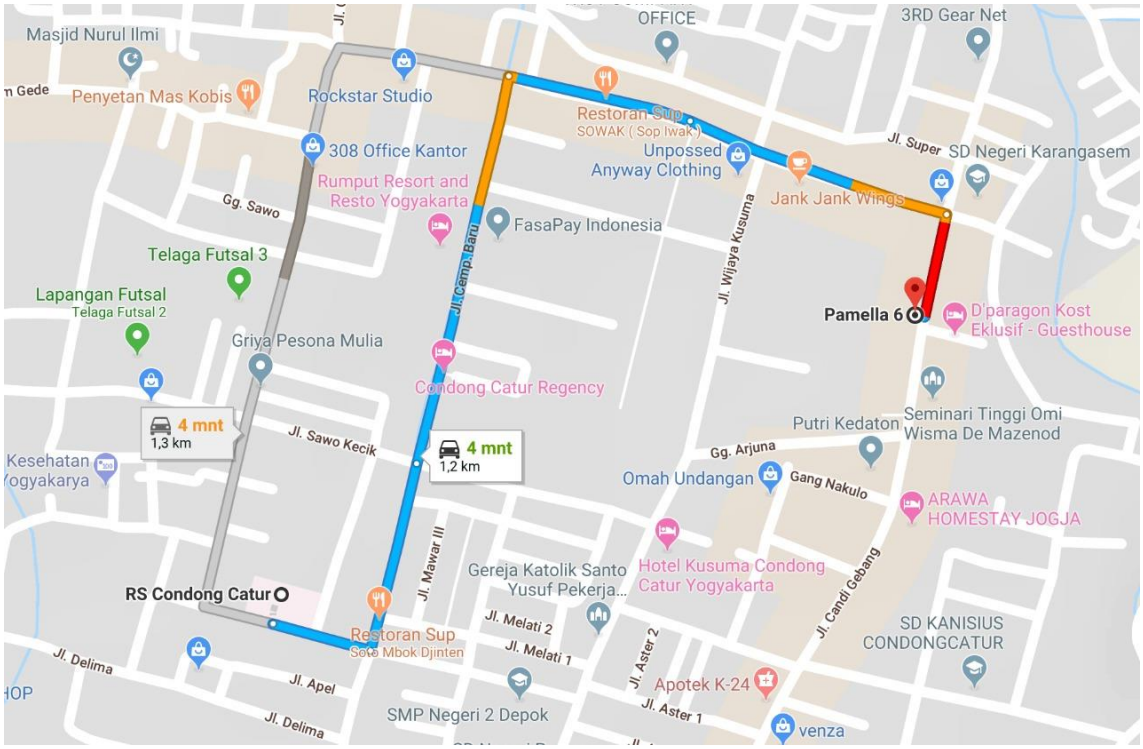
Judul Paper	Metode/Algoritma	Variabel Data	Objek	Perbedaan
MENGHITUNG RUTE TERPENDEK MENGGUNAKAN ALGORITMA A* DENGAN FUNGSI EUCLIDEAN DISTANCE	Astar dengan heuristik euclidean	Titik(node) yang terdapat dalam dekat universitas negeri jakarta	SIG(Google Maps)	Penelitian menggunakan 2 algoritma,hasil penelitian adalah perbandingan waktu,jumlah langkah dan konsumsi cpu dan ram
PENCARIAN RUTE TERPENDEK TEMPAT WISATA DI BALI DENGAN MENGGUNAKAN ALGORITMA DIJKSTRA	DIJKSRA	TEMPAT WISATA DI BALI	SIG (Google Maps)	Penelitian menggunakan 2 algoritma,hasil penelitian adalah perbandingan waktu,jumlah langkah dan konsumsi cpu dan ram
Rancang Bangun Aplikasi untuk Menentukan Jalur	Algoritma Dijkstra	Data primer Jarak yang diteliti yaitu	SIG	Penelitian menggunakan 2 algoritma,hasil

Terpendek Rumah Sakit di Purbalingga dengan Metode Algoritma Dijkstra	dari Alun-alun Purbalingga ke berbagai Rumah Sakit di Purbalingga.	penelitian adalah perbandingan waktu,jumlah langkah dan konsumsi cpu dan ram

2. METODE PENELITIAN

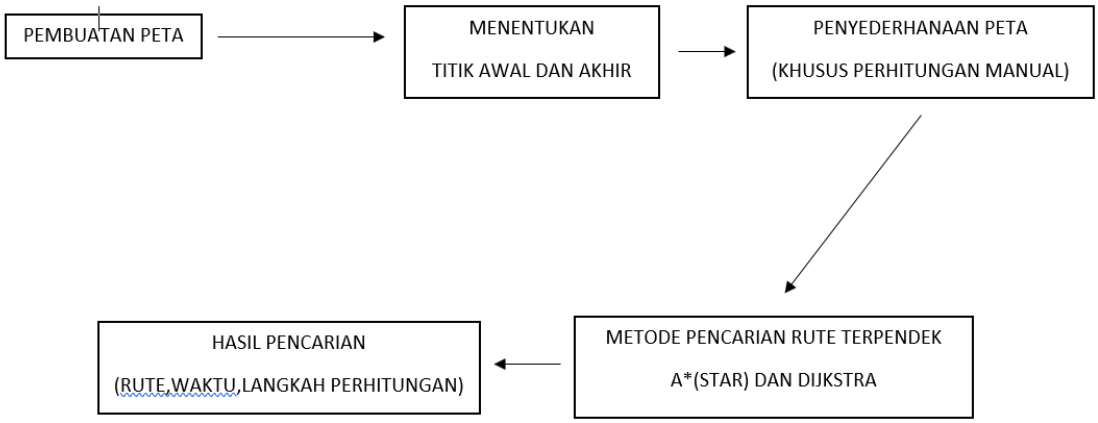
2.1 Perancangan

Penelitian menggunakan tampilan dari Google Maps dengan cakupan daerah Condong Catur, Depok, Sleman, Yogyakarta. Sebelumnya telah ditentukan untuk *Start node* atau titik awal yaitu Pasar Swalayan Pamela 6. Sedangkan untuk *Destination* atau lokasi tujuan adalah Rumah Sakit Condong Catur. Di sepanjang node awal dan lokasi tujuan, telah ditentukan node-node yang fungsinya untuk menandai simpangan-simpangan jalan yang memungkinkan untuk dilewati



Gambar 1. Tampilan Google Maps Daerah Condong Catur, Depok, Sleman

Tampilan Google Maps ini digunakan sebagai media atau objek untuk membuktikan kinerja dari Algoritma A* dan Algoritma Dijkstra. Penelitian mengggunakan lokasi tersebut karena daerahnya yang strategis, dimana memiliki beberapa opsi untuk dilewati karena banyaknya persimpangan dan jarak kedua tempat yang tidak terlalu jauh namun juga tidak terlalu dekat.



Gambar 2. Alur Penelitian

Langkah pertama adalah pembuatan peta yang telah disederhanakan dari google maps, lengkap dengan titik awal dan titik tujuan akhirnya.selanjutnya perhitungan secara teoritis pada kedua algoritma akan dibahas,algoritma tersebut akan diterapkan dalam aplikasi. Setelah itu kedua algoritma akan dibandingkan pada aplikasi,variable yang diuji dibatasi pada jumlah langkah,waktu dan rute.

2.2 Sistem Informasi Geografis (SIG/GIS)

Sistem Informasi Geografis(SIG/GIS) merupakan system informasi berbasis computer [9] yang dapat mengintegrasikan berbagai operasi seperti basis data query dan pencarian rute terdekat/terbaik, menganalisisnya serta menampilkannya dalam bentuk peta/graph [10]

2.3 Google Maps

Google Maps adalah sebuah jasa peta digital gratis dan online yang disediakan oleh Google. Google Maps dapat diakses oleh semua orang melalui alamat URL http://maps.google.com dari segala perangkat**.** Google Maps menawarkan peta yang dapat diseret dari satelit untuk seluruh dunia.

Baru-baru ini Google telah meluncurkan fitur baru untuk Google Maps, yaitu Maps GL dimana fitur ini memberikan penyempurnaan dari segi grafis, dan akses yang lebih mudah ke Street View.

2.4 Algoritma Pencarian rute Terdekat (Pathfinding Algorithm)

Menurut KBBI Algoritma atau Algoritme adalah urutan logis pengambilan keputusan untuk pemecahan masalah.dalam pemrograman algoritma merupakan hal yang sangat penting,jika terjadi kesalahan sedikit saja program tidak akan berjalan.Oleh karena itu dibutuhkan algoritma yang baik dalam menyelesaikan masalah sehingga program dapat berjalan dengan baik.salah satu permasalahan yang membutuhkan penyusunan algoritma yang baik adalah pencarian rute terdekat(Pathfinding).Algoritma Pencarian rute terpendek dibagi menjadi 2 yaitu:

Pencarian Buta atau Konvensional adalah pencarian dari setiap kemungkinan yang ada dari permasalahan [11] dengan tujuan menguji semua kemungkinan yang ada untuk menemukan solusinya [12].Contoh dari algoritma pencarian buta adalah Dijkstra.kelemahan dari pencarian ini adalah waktu yang dibutuhkan lebih lama dan jumlah langkahnya lebih banyak karena Pencarian ini akan melihat setiap kemungkinan

Pencarian Heuristik adalah bidang dari Kecerdasan Buatan (Artificial Intelligence)[12] dengan menggunakan Pencarian ini,waktu yang dibutuhkan lebih sedikit dan jumlah langkahnya lebih sedikit.Hal ini disebabkan oleh pencarian memprioritaskan kemungkinan terbaik sehingga

solusi lebih cepat ditemukan. Contoh dari Heuristik yang popular digunakan pada algoritma astar adalah Manhattan, Euclidean,Octile dan Chebysev.[13]

2.5 Teorema Pitagoras

Teorima pitagoras digunakan untuk menentukan Panjang dari sisimiring sebuah segitu siku-siku.teori ini hanya berlaku pada segitiga siku-siku dan dapat digunakan jika Panjang dan tinggi segitiga telah diketahui[14]. Persamaan dari teorema pitagoras adalah :

$$\boldsymbol{sisi\ miring = \sqrt{panjang^2 + tinggi^2}}\ \textit{(1)}$$

2.6 Konsep Algoritma A*

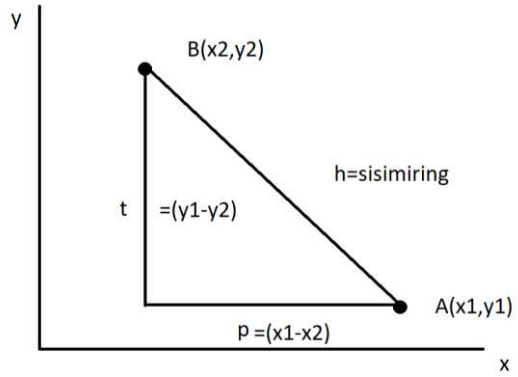
Algoritma A* dikembangkan oleh Hart, Nilsson, dan Raphael, algoritma ini paling popular dan telah digunakan secara luas dalam graph traversal, penemuan jalur, serta proses perencanaannya .[2]

Open List adalah titik yang boleh dilalui contohnya,titik yang belum dilalui dan titik tersebut bukanlah tembok,close list adalah titik yang tidak boleh dilalui misalnya tiitk tersebut adalah tembok,dan titik tersebut sudah dilalui,fungsi dari close list salah satunya adalah agar algoritma tersebut tidak mengecek lagi titik yang sudah dilalui sehingga pencarian lebih cepat dan tidak ada perulangan tak terbatas,biasanya algoritma pencarian rute terpendek akan berhenti jika,tidak ada lagi open list atau titik akhir sudah ditemukan [6].

Graph adalah kumpulan titik(vertex/node) yang dihubungkan dengan garis(edge) dengan titik lainnya[8].nilai graph pada astar dapat dihitung dengan menjumlahkan langkah yang diperlukan untuk bergerak dari satu titik ke titik lainnya.

Metode heuristik yang digunakan adalah Euclidean**.** Rumus Euclidean dipilih karena rumus ini mempunyai dasar teori yang kuat yaitu Pitagoras. Selain itu rumus ini juga cukup mudah digunakan..

dalam mencari titik terdekat A ke B maka ditariklah garis lurus A ke B, selanjutnya heuristik dianggap sebagai sisimiring pada segitiga siku-siku, segitiga dibentuk dari 2 sumbu yaitu x dan y, maka didapatlah persamaan sebagai berikut :



Gambar 3. Ilustrasi pitagoras

Dari gambar ilustrasi diatas, digunakan rumus sebagai berikut :

$$\boldsymbol{sisi\ miring\ (h) = \sqrt{panjang^2 + tinggi^2}}\ \textit{(1)}$$

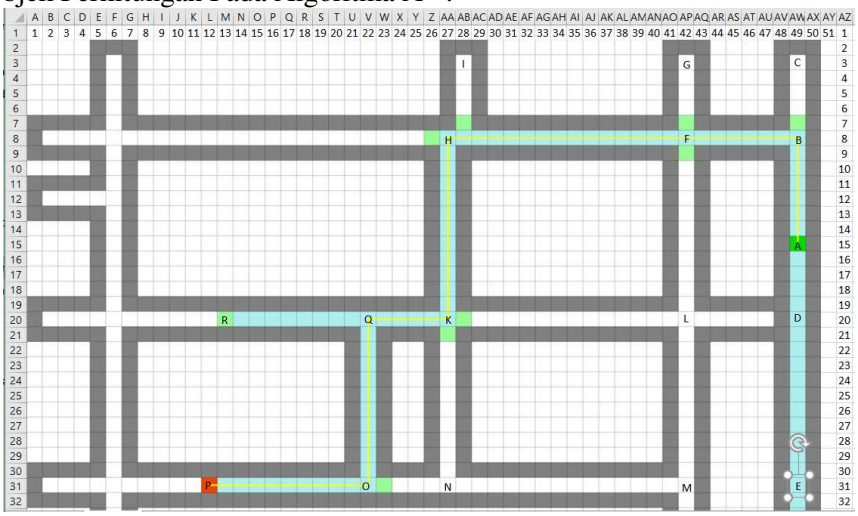
Persamaan tersebut masih belum bisa digunakan karena Panjang dan tinggi segitiga belum diketahui, oleh karena itu persamaan tersebut dijabarkan lagi sehingga nilai Panjang dan tinggi dapat diketahui.

$$\boldsymbol{h = \sqrt{(x1 - x2)^2 + (y1 - y2)^2}} \qquad \textbf{(2)}$$

Algoritma A* merupakan penyempurnaan dari Algoritma Dijkstra dan Membentuk Algoritma Best First Search (BFS) paling pertama.Algoritma A* akan bekerja dengan menjumlahkan g(n), yaitu jumlah pergerakan dari satu node ke node lain dan h(n) yaitu perkiraan biaya dari node n ke tempat tujuan akhir. Sehingga di dapatkan Persamaan [2]:

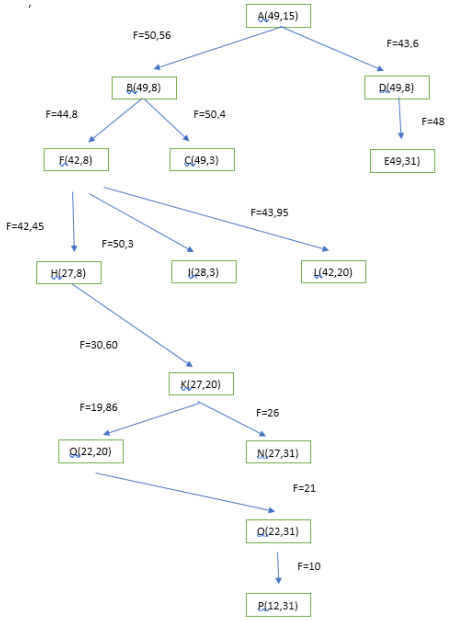
$$\boldsymbol{f(n) = g(n) + h(n)} \qquad \textbf{(3)}$$

$\boldsymbol{f(n)}$ = total biaya (jarak) yang diperlukan untuk jalan dari satu node ke tujuan.
$\boldsymbol{h(n)}$ = perkiraan biaya (jarak) dari node n ke tujuan akhir (heuristik).
$\boldsymbol{g(n)}$ = jumlah pergerakan dari satu node ke node lain.(graph)
Tampilan Objek Perhitungan Pada Algoritma A* :



Gambar 4. Ilustrasi Rute A star

Hitung nilai F dari masing-masing titik. Ambillah titik dengan nilai F terendahnya, selanjutnya hitunglah cabangnya (titik didekatnya). Ulangi langkah tersebut hingga ditemukan titik akhir, selanjutnya untuk memudahkan perhitungan maka dibuatlah pohon keputusan :



Gambar 5. Pohon Keputusan Astar

Setelah pohon keputusan terbentuk semua selanjutnya kita akan menentukan rute dengan nilai F terkecil sehingga didapat rute **A-B-F-H-K-Q-O-P.**

Pseudocode Astar :

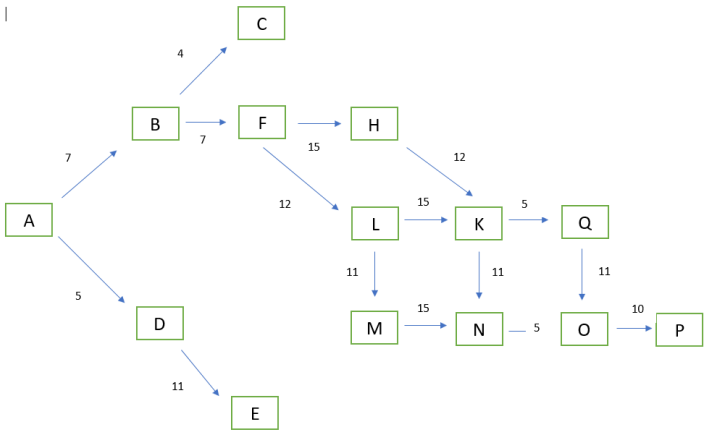
```
// A* (star) Pathfinding
// Initialize both open and closed list
let the openList equal empty list of nodes
let the closedList equal empty list of nodes
// Add the start node
put the startNode on the openList (leave it's f at zero)
// Loop until you find the end
while the openList is not empty
    // Get the current node
    let the currentNode equal the node with the least f value
    remove the currentNode from the openList
    add the currentNode to the closedList
    // Found the goal
    if currentNode is the goal
        Congratz! You've found the end! Backtrack to get path
    // Generate children
    let the children of the currentNode equal the adjacent nodes

    for each child in the children
        // Child is on the closedList
        if child is in the closedList
            continue to beginning of for loop
        // Create the f, g, and h values
        child.g = currentNode.g + distance between child and current
        child.h = distance from child to end
        child.f = child.g + child.h
        // Child is already in openList
        if child.position is in the openList's nodes positions
            if the child.g is higher than the openList node's g
                continue to beginning of for loop
        // Add the child to the openList
        add the child to the openList
```

Gambar 6. Pseudocode Algoritma A*

2.7 Konsep Algoritma Djikstra

Algoritma Djikstra dibuat oleh Edsger Dijkstra. Dijkstra digunakan untuk mencari rute terpendek [4]. Algoritma Dijkstra termasuk algoritma rakus (greedy) yang dapat digunakan ketika graph mempunyai garis(edge) yang berbobot positif dan mempunyai arah [5]. Algoritma ini bekerja dengan mengunjungi semua titik yang ada dan membuat rutenya jika ada 2 rute menuju 1 titik yang sama maka rute yang memiliki bobot terendahlah yang dipilih sehingga semua titik mempunyai rute yang optimal. Pencarian ini berlangsung sampai titik tujuan terakhir telah ditemukan. Kelemahan dari algoritma ini adalah tidak terpusatnya pencarian,sehingga proses pencarian akan berlangsung lama dan mempunyai jumlah langkah yang besar.



Gambar 8. Penyederhanaan Graph

Graph yang digunakan adalah graph dari algoritma astar,namun graph tersebut disederhanakan sehingga perhitugan dapat dilakukan dengan mudah dan lebih singkat.Penyerdahaan graph ini tidak akan mengubah hasil rutenya.

Tabel 2. Tabel perhitungan Algoritma Dijkstra

A	B	C	D	E	F	G	H	I	J
0	infinite	infinite	infinite	infinite	infinite	infinite	infinite	infinite	infinite
0	A-B=7	infinite	A-D=5	infinite	infinite	infinite	infinite	infinite	infinite
0	A-B=7	A-B-C=4	A-D=5	A-D-E=11	A-B-F=14	infinite	infinite	infinite	infinite
0	A-B=7	A-B-C=4	A-D=5	A-D-E=11	A-B-F=14	A-B-F-G=19	A-B-F-H=29	infinite	infinite
0	A-B=7	A-B-C=4	A-D=5	A-D-E=11	A-B-F=14	A-B-F-G=19	A-B-F-H=29	infinite	infinite

K	L	M	N	O	P	Q
A-B-F-H-K=41 A-B-F-L-K=41	A-B-F-L=26	A-B-F-L-M=37	infinite	infinite	infinite	infinite
A-B-F-H-K=41 A-B-F-L-K=41	A-B-F-L=26	A-B-F-L-M=37	A-B-F-L-M-N=52 A-B-F-H-K-N=52 A-B-F-L-K-N=52	infinite	infinite	A-B-F-H-K-Q=46 A-B-F-L-K-Q=46
A-B-F-H-K=41 A-B-F-L-K=41	A-B-F-L=26	A-B-F-L-M=37	A-B-F-L-M-N=52 A-B-F-H-K-N=52 A-B-F-L-K-N=52	A-B-F-L-M-N-O=57 A-B-F-H-K-N-O=57 A-B-F-L-K-N-O=57 A-B-F-H-K-Q-O=57 A-B-F-L-K-Q-O=57	infinite	A-B-F-H-K-Q=46 A-B-F-L-K-Q=46
A-B-F-H-K=41 A-B-F-L-K=41	A-B-F-L=26	A-B-F-L-M=37	A-B-F-L-M-N=52 A-B-F-H-K-N=52 A-B-F-L-K-N=52	A-B-F-L-M-N-O=57 A-B-F-H-K-N-O=57 A-B-F-L-K-N-O=57 A-B-F-H-K-Q-O=57 A-B-F-L-K-Q-O=57	infinite	A-B-F-H-K-Q=46 A-B-F-L-K-Q=46
A-B-F-H-K=41 A-B-F-L-K=41	A-B-F-L=26	A-B-F-L-M=37	A-B-F-L-M-N=52 A-B-F-H-K-N=52 A-B-F-L-K-N=52	A-B-F-L-M-N-O=57 A-B-F-H-K-N-O=57 A-B-F-L-K-N-O=57 A-B-F-H-K-Q-O=57 A-B-F-L-K-Q-O=57	A-B-F-L-M-N-O-P=67 A-B-F-H-K-N-O-P=67 A-B-F-L-K-N-O-P=67 A-B-F-H-K-Q-O-P=67 A-B-F-L-K-Q-O-P=67	A-B-F-H-K-Q=46 A-B-F-L-K-Q=46

Pertama inisialisasikan semua titik nilainya tak hingga, selanjutnya set titik awal nilainya adalah 0, lalu kunjungi titik didekatnya dan tulis biaya perjalanan pada titik didekatnya, Jika ada 2 rute menuju 1 titik yang sama pilihlah rute yang memiliki nilai terendahnya, dan simpan hasil nilai terendahnya, ulangi langkah tersebut sampai ditemukan titik akhirnya. Selanjutnya carilah rute yang menuju titik akhir dengan memilih nilai rute terendahnya saja di tiap titik didekatnya, rute akhirnya akan optimal karena di titik sebelumnya telah disimpan nilai optimalnya saja.

```
function Dijkstra(Graph, source):
    for each vertex v in Graph:              // Initialization
        dist[v] := infinity                  // initial distance from source to vertex v is set to infinite
        previous[v] := undefined             // Previous node in optimal path from source
    dist[source] := 0                        // Distance from source to source
    Q := the set of all nodes in Graph       // all nodes in the graph are unoptimized - thus are in Q
    while Q is not empty:                    // main loop
        u := node in Q with smallest dist[ ]
        remove u from Q
        for each neighbor v of u:            // where v has not yet been removed from Q.
            alt := dist[u] + dist_between(u, v)
            if alt < dist[v]                 // Relax (u,v)
                dist[v] := alt
                previous[v] := u
    return previous[ ]
```

Gambar 9. Pseudocode Algoritma Dijkstra.

3. HASIL DAN PEMBAHASAN

3.1. Percobaan

Percobaan dilakukan dengan bantuan software library *PathFinding.js* yang berisi *code* untuk melakukan pencarian rute terdekat dengan Algoritma A* dan Algoritma Dijkstra. Pada software tersebut diinputkan node awal, node akhir, serta node-node atau persimpangan yang sebelumnya telah ditandai pada Google Maps.

Pada percobaan ini, peneliti menggunakan perangkat dengan spesifikasi Intel Core i5, CPU 2.7 GHz, 8 GB RAM. Percobaan dilakukan sebanyak 10 kali, agar data yang didapat cukup akurat. Setelah software library *PathFinding.js* diinputkan titik (node) awal dan akhir pada peta labirin yang tampilannya disesuaikan dengan tampilan peta google maps, didapatkan tampilan rute yang berbeda dari kedua Algoritma.

Perbedaan tersebut disebabkan oleh masing-masing algoritma menerapkan dua mekanisme yang berbeda. Dari percobaan tersebut, ternyata rute dari Algoritma A* adalah rute yang paling mendekati metode pencarian yang dilakukan Google Maps itu sendiri.

3.2. Hasil Percobaan

Penelitian diulang sebanyak 10 kali percobaan untuk mencari nilai rata-rata dari setiap algoritma sehingga data yang dihasilkan akan lebih valid. Percobaan dilakukan untuk membandingkan kinerja kedua algoritma dalam segi waktu pemrosesan dan resource (sumber daya) komputer pada pencarian rute terdekat.

Perhitungan nilai tidak dilakukan secara manual, melainkan dengan bantuan dari software penguji itu sendiri yaitu *PathFinding.js* sehingga data yang dihasilkan lebih berbobot. Dari percobaan ini, di dapatkan hasil :

Tabel 3. Tabel Dijkstra

Dijkstra			
No	**Time (ms)**	**Memory (MB)**	**CPU(%)**
1	0.8000	90	37.6
2	0.2000	92.8	19.3
3	0.2000	91.7	22.3
4	0.3000	91.1	27.8
5	0.6000	91.1	12.6
6	0.2000	88.1	9.5
7	0.6000	90.3	19.2
8	0.3000	88.1	8.3
9	0.7000	91	10.1
10	0.2000	92.1	20.8
Rata-Rata	0.41	90.63	18.75

Tabel 4. Tabel Astar

A* Euclidean			
No	**Time (ms)**	**Memory (MB)**	**CPU (%)**
1	0.2000	90	14.1
2	0.3000	89.5	13.3
3	0.8000	89.6	16.4
4	0.8000	88.8	3.5
5	0.6000	80.9	10.4

6	0.2000	88.9	7.2
7	0.3000	89.3	2.2
8	0.2000	91	7.5
9	0.1000	88	0
10	0.2000	89.3	2.8
Rata-Rata	0.37	88.53	7.74

Tabel 5. perbandingan jumlah langkah

Jenis Algoritma	Jumlah langkah
Algoritma A*	200
Algoritma Dijkstra	497

Tabel 6. Rute

Jenis Algoritma	Rute	keterangan
Algoritma A*	A-B-F-H-K-Q-O-P	Optimal(Rute Terdekat)
Algoritma Dijkstra	A-B-F-L-M-N-O-P	Optimal(Rute Terdekat)

4. KESIMPULAN

Dari percobaan ini, dengan objek dari node mulai dan node tujuan yang sama, disimpulkan bahwa Algoritma A* dapat menentukan waktu proses pencarian rute terdekat dengan lebih cepat dengan rata-rata waktu 0.37 ms dan langkah 200 daripada Algoritma Dijkstra yaitu 0.41 ms dan langkah 497. Sehingga didapatkan kesimpulan lain, yaitu Algoritma A* bekerja lebih baik dan lebih efisien daripada Algoritma Dijkstra. Selain itu, data membuktikan bahwa waktu yang dibutuhkan Algoritma A* lebih stabil dari pada Algoritma Dijkstra. Dengan menggunakan kombinasi nilai g dan h maka Strategi yang digunakan algoritma A*(Star) cukup ampuh untuk diterapkan pada persoalan-persoalan pencarian rute terpendek. Hasil solusi yang didapatkan oleh A* mendekati menghampiri hasil solusi optimal.

5. SARAN

Masih banyak metode atau algoritma dalam menentukan algoritma pencarian rute terpendek contohnya Best First Search, Breadth First Search, IDA*, Jump Point Search. Dalam menentukan rute terbaik dan tercepat bukan hanya jarak terdekat saja yang diperhitungkan, Diperlukan variable lain seperti padatnya jalan sehingga dapat diprediksi perkiraan waktu yang ditempuh.

6. DAFTAR PUSTAKA

[1] Saputra, A.J. *Penerapan Algoritma A* pada Google Map*. Teknik Informatika. Institut Teknologi Bandung. 2013.

[2] Setiawan, K., Supriyadin, Santoso, I., Buana, R. *Menghitung Rute Terpendek Menggunakan Algoritma A* Dengan Fungsi Euclidean Distance*. Teknik Informatika. Sekolah Tinggi Ilmu Komputer Cipta Karya Informatika. 2018.

[3] Munir, R., Matematika Diskrit, 4, 413-414, Informatika, Bandung, 2010.

[4] Dewi, L.J.E. *Pencarian Rute Terpendek Tempat Wisata Di Bali Dengan Menggunakan Algoritma Dijkstra.* Fakultas Teknik dan Kujuruan. Universitas Pendidikan Ganesha. 2010.

[5] Wibowo, A.G., Wicaksono, A.P. *Rancang Bangun Aplikasi untuk Menentukan Jalur Terpendek Rumah Sakit di Purbalingga dengan Metode Algoritma Dijkstra*. Teknik Informatika. Universitas Muhammadiyah Purwokerto. 2012.

[6] Yenie Syukriyah, Falahah, Hermi Solihin Penerapan Algoritma A* (Star) Untuk Mencari Rute Tercepat Dengan Hambatan. Seminar Nasional Telekomunikasi dan Informatika 2016, BANDUNG 28 MEI 2016

[7] Masri., Mukti, A.T. *Pencarian Jalur Terpendek Pada Snake Game Menggunakan Algoritma A**. Sekolah Tinggi Manajemen Informatika dan Komputer Pontianak. SISFOTENIKA. 2014.

[8] Lely Hiryanto, Jacklin Sinthia Thio Pengembangan Metode Graph Coloring Untuk University Course Timetabling Problem Pada Fakultas Teknologi Informasi Universitas Tarumanagara. Laboratorium Penelitian Distributed System, Fakultas Teknologi Informasi, Universitas Tarumanagara

[9] Kevin Bima Aditya, Diyah P, Yudi Setiawan Sistem Informasi Geografis Pemetaan Faktor-Faktor Yang Mempengaruhi Angka Kematian Ibu (Aki) Dan Angka Kematian Bayi (Akb) Dengan Metode K-Means Clustering (Studi Kasus: Provinsi Bengkulu) Program Studi Teknik Informatika, Fakultas Teknik, Universitas Bengkulu

[10] M. Akbar Maulana, Andi Kriswantoro, Yans Safarid Hudha, Muhammad Habib, Syarham, Ema Utami, Sistem Informasi Geografis Pemetaan Praktik Dokter Umum Dan Spesialis ―Smart Doctor" 105. CSRID Journal, Vol.9 No.2 Juni 2017

[11] Novriyanto,M Zaid Penerapan Algoritma Backtracking Berbasis Blind Search untuk menentukan Penjadwalan Mengajar Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Sultan Syarif Kasim Riau

[12] Iing Mutakhiroh, Fajar Saptono, Nur Hasanah, Romi Wiryadinata Pemanfaatan Metode Heuristik Dalam Pencarian Jalur Terpendek Dengan Algoritma Semut Dan Algoritma Genetika Laboratorium Pemrograman dan Informatika Teori, Universitas Islam Indonesia

[13] Rudy Adipranata , Andreas Handojo , Happy Setiawan Aplikasi Pencari Rute Optimum Pada Peta Guna Meningkatkan Efisiensi Waktu Tempuh Pengguna Jalan Dengan Metode A* Dan Best First Search Jurnal Informatika VOL. 8, NO. 2, NOPEMBER 2007: 100 – 108

[14] Wahyudin Djumantra Mari Memahami Konsep Matematika buku pelajaran matematika untuk kelas viii grafindo media pratama cetakan 1 2005 Bandung