Jurnal Widya
Volume 3, Nomor 1, April 2022: halaman 59-67
https://jurnal.amikwidyaloka.ac.id/index.php/awl
jurnal@amikwidyaloka.ac.id / editor.jurnalwidya@gmail.com

P-ISSN: 2746-5411
e-ISSN: 2807-5528

ANALISIS PERBANDINGAN ALGORITMA DFS, BFS DAN DIJKSTRA UNTUK MENENTUKAN RUTE TERPENDEK PADA PETA GEOGRAFIS

Fenisa Lourence Tobing[1], Fenina Adline Twince Tobing[2*], Prayogo[3]
[1]Program Studi Manajemen Informatika, AMIK Widya Loka, Medan, Sumatera Utara, Indonesia
[2]Program Studi Informatika, Fakultas Teknik dan Informatika Universitas Multimedia Nusantara, Tangerang, Banten, Indonesia
[3]Program Studi Teknik Informatika, AMIK Mapan, Tangerang, Banten, Indonesia

Email : [1]*fenishatobing@gmail.com* [2]*fenina.tobing@umn.ac.id*

Received: January 30, 2022. **Revised:** February 28, 2022. **Accepted:** March 26, 2022.
DOI : https://doi.org/10.54593/awl.v3i1.83

Abstrak

Ada berbagai algoritma yang akan diterapkan yang dapat memudahkan setiap aktivitas manusia. Dalam pencarian jalur terpendek, terdapat tiga algoritma yang cukup dikenal sebagai pencari jalur terpendek yaitu DFS, BFS dan Dijkstra. Ketiga metode tersebut dibandingkan untuk mengetahui algoritma mana yang paling efisien dan sesuai dalam penelitian ini. Penelitian ini memberikan informasi mengenai algoritma mana yang cocok untuk diterapkan dalam pencarian rute terpendek berdasarkan peta geografis. Hasil dari perbandingan ini adalah perhitungan berupa jarak tempuh antar node yang didekati sesuai dengan karakteristik masing-masing algoritma dan urutan alogoritma dalam pencarian jarak terpendek berdasarkan peta geografis adalah Algoritma Dijkstra, Algoritma DFS dan Algoritma BFS.

Kata kunci : Algoritma, DFS, BFS, Dijkstra, Rute, Peta.

Abstract

There are various algorithms that will be implemented that can facilitate every human activity. In searching for the shortest path, there are three algorithms that are well known as the shortest path finder, namely DFS, BFS and Dijkstra. The three methods are compared to find out which algorithm is the most efficient and appropriate in this study. This research provides information about which algorithm is suitable to be applied in finding the shortest route based on geographic maps. The result of this comparison is the calculation of the distance between nodes that are approached according to the characteristics of each algorithm and the order of the algorithms in searching for the shortest distance based on geographic maps is Dijkstra's Algorithm, DFS Algorithm and BFS Algorithm.

Keywords ***:*** *Algorithm, DFS, BFS, Dijkstra, Route, Map*

1 Pendahuluan (Introduction)

Peta adalah gambaran permukaan bumi dengan menggunakan skala tertentu yang digambar pada bidang datar melalui sistem proyeksi tertentu [1]. Peta tidak pernah termakan oleh waktu, meski di era digital, keberadaan peta tetap menjadi kebutuhan bagi sebagian besar orang. Saat ini banyak aplikasi digital yang mampu mengelola peta yang dapat diakses melalui suatu perangkat sehingga dapat

Jurnal Widya
Volume 3, Nomor 1, April 2022: halaman 59-67
https://jurnal.amikwidyaloka.ac.id/index.php/awl
jurnal@amikwidyaloka.ac.id / editor.jurnalwidya@gmail.com
P-ISSN: 2746-5411
e-ISSN: 2807-5528

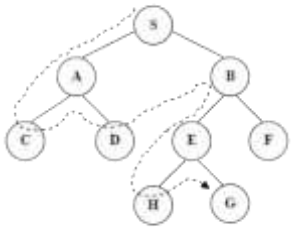
diakses oleh siapa saja. Sama seperti pada peta lama, peta digital masih digunakan untuk menuju suatu tempat. Dan untuk menuju suatu tempat yang pasti dibutuhkan jalur terpendek. Namun butuh waktu dan ketelitian untuk menemukan jalur terpendek di peta.

Untuk menentukan jalur terpendek antar tujuan digunakan beberapa metode yaitu Depth First Search (DFS), Breadth First Seacrh (BFS) dan Dijkstra. Menurut A Goldberg, peneliti Microsoft Research Silicon Valley, mengatakan ada banyak alasan mengapa peneliti terus mempelajari masalah menemukan jalur terpendek. "Jalur terpendek adalah masalah optimasi yang relevan untuk berbagai aplikasi, seperti perutean jaringan, permainan, desain sirkuit, dan pemetaan" [2]. Beberapa dari metode ini terkenal karena fungsinya untuk mencapai jalur terdekat dalam konsep graf. Pada penelitian ini akan dibandingkan metode algoritma yang paling efisien untuk mencari jalur terpendek dari ketiga algoritma tersebut.

Dalam penerapannya, proses pencarian rute terpendek menggunakan sistem yang disebut sistem informasi geografis (SIG). SIG adalah sistem yang dirancang untuk menangkap, menyimpan, memanipulasi, menganalisis, mengatur dan menampilkan semua jenis data geografis [6]. Ketiga algoritma yang akan diuji dapat dihubungkan dan bekerja sama untuk menentukan jalur terpendek. Kajian ilmiah ini akan menguji algoritma mana yang paling cepat dan efisien untuk menentukan jalur terpendek yang mengambil studi kasus dari kota Semarang.

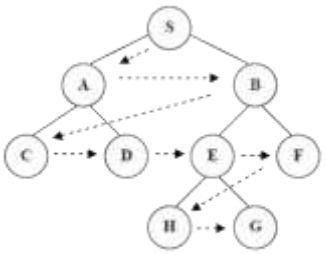
2 Tinjauan Literatur

Menurut Derwin Suhartono, algoritma DFS adalah teknik pencarian dengan melakukan ekspansi ke arah node terdalam dalam sebuah pohon [3]. Node terdalam ditandai dengan tidak adanya penerus dari node tersebut. Setelah node diekspansi, node tersebut akan ditinggalkan dan dibawa ke node terdalam lainnya yang masih memiliki penerus yang belum diekspansi.



Gambar 1. Pohon DFS

Menurut Derwin Suhartono, algoritma BFS adalah pencarian dengan menggunakan teknik dimana langkah pertama adalah memperluas root node, setelah itu semua penerus dari root node juga diperluas [4]. Hal ini terus dilakukan berulang-ulang hingga node yang paling bawah tidak memiliki penerus lagi



Gambar 2. Pohon BFS

Jurnal Widya
Volume 3, Nomor 1, April 2022: halaman 59-67
https://jurnal.amikwidyaloka.ac.id/index.php/awl
jurnal@amikwidyaloka.ac.id / editor.jurnalwidya@gmail.com
P-ISSN: 2746-5411
e-ISSN: 2807-5528

Menurut uraian Abba Suganda Girsang, "Algoritma Dijkstra adalah algoritma yang digunakan dalam menyelesaikan masalah jalur terpendek untuk graf berarah" [5]. Algoritma Dijkstra bekerja dengan membuat jalur ke satu node optimal pada setiap langkah. Jadi pada langkah n, setidaknya ada n node yang sudah kita ketahui sebagai jalur terpendek

Gambar 3. Grafik Dijkstra

3 Metode Penelitian (or Research Method)

Hampir setiap orang membutuhkan jalur terpendek untuk mencapai suatu tujuan. Efisiensi waktu dan menekan pengeluaran bensin adalah salah satu dari dua faktor lain mengapa jalur terpendek untuk mencapai tujuan menjadi penting. Tidak terlepas dari kota Semarang, Semarang merupakan kota yang merupakan ibu kota provinsi Jawa Tengah. Dengan luas kota 373,78 km2 dengan jumlah penduduk 1.555.984 jiwa dan kepadatan 4.200/km2, tidak heran jika kota Semarang termasuk kota yang cukup padat. Kemacetan menjadi hal yang tak terhindarkan. Sehingga penduduk kota Semarang membutuhkan alternatif untuk menghindari kemacetan yang membuat perjalanan menjadi tidak efisien. Aplikasi peta digital (Waze, Google Maps) menjadi pilihan warga untuk mempersingkat perjalanan.

Pada penelitian ini rute diambil dari titik awal di Heritage, Kota Lama menuju Gang Pasar Baru. Rute tersebut diambil sebagai salah satu contoh untuk mengetahui jalur tercepat dari beberapa pilihan jalan yang ada.

Gambar 4. Peta Kota Semarang

Dari titik A ke titik J sebagai endpoint tujuan dapat melewati beberapa titik lokasi yang akan dianalogikan sebagai node agar lebih mudah. Dijelaskan sebagai berikut:

Jurnal Widya
Volume 3, Nomor 1, April 2022: halaman 59-67
https://jurnal.amikwidyaloka.ac.id/index.php/awl
jurnal@amikwidyaloka.ac.id / editor.jurnalwidya@gmail.com
P-ISSN: 2746-5411
e-ISSN: 2807-5528

Gambar 5. Peta Kota Semarang (dengan simpul)

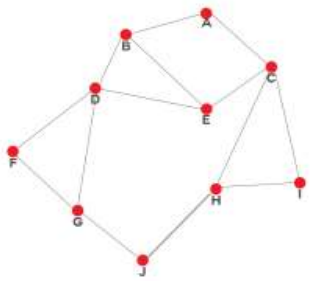
Berdasarkan peta tersebut dibuat tabel sebagai berikut:

Node	Nama tempat	Jarak tempuh dari titik awal (m)	Estimasi waktu (menit)
A	Heritage Kota Lama	0	0
B	Het Groote Huis	350	1
C	Toko Besi New Lestari	900	3
D	New Metro	850	2
E	Toko Plastik Tan	400	2
F	Toko Oen	1000	3
G	Toko Tekstil Kranggan	1400	4
H	Semawis Market	750	3
I	Surya Jaya Sport	2500	6
J	Pasar Gang Baru	1400	5

1. **Algoritma DFS**

Algoritma ini melakukan pencarian secara mendalam pada semua node dengan terus melakukan pencarian ke bawah hingga belum menemukan ujung graf.

Jika digambarkan dalam grafik akan menjadi seperti ini:



Gambar 6. Grafik DFS (sesuai peta)

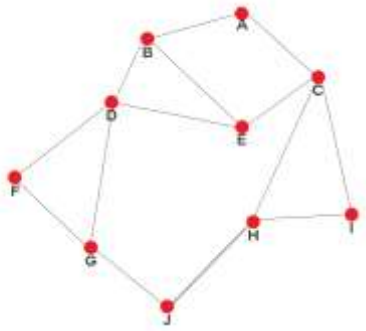
Dengan langkah-langkah berikut :

1. Masukkan simpul ke dalam tumpukan.
2. Simpan elemen dari tumpukan teratas.
3. Hapus isi tumpukan atas dengan prosedur pop.
4. Periksa simpul pohon yang disimpan (periksa simpul anak).
5. Jika demikian, masukkan semua simpul ke dalam tumpukan.
6. Jika tumpukan kosong berhenti tetapi jika tidak maka kembali ke langkah 2.

Jurnal Widya
Volume 3, Nomor 1, April 2022: halaman 59-67
https://jurnal.amikwidyaloka.ac.id/index.php/awl
jurnal@amikwidyaloka.ac.id / editor.jurnalwidya@gmail.com
P-ISSN: 2746-5411
e-ISSN: 2807-5528

2. **Algoritma BFS**

Algoritma ini hampir sama dengan DFS tetapi algoritma BFS melakukan pencarian secara menyeluruh, kemudian menelusuri semua node di bawahnya.



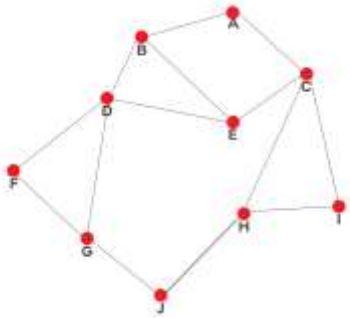
Gambar 7. Grafik BFS (sesuai peta)

Dengan langkah-langkah berikut:

1. Masukkan node dalam antrian.
2. Periksa antrian depan (periksa simpul anak).
3. Jika benar, letakkan semua node di tumpukan.
4. Hapus antrian terdepan.
5. Jika antrian kosong maka berhenti. Tetapi jika tidak kembali ke langkah 2.

3. **Algoritma Dijkstra**

Algoritma ini dikenal sebagai algoritma serakah. Cara kerjanya adalah dengan membuat path ke salah satu node yang sudah terpendek.



Gambar 8. Grafik Dijkstra (menurut peta)

Dengan langkah-langkah berikut:

1. Tentukan titik mana yang akan menjadi simpul awal, kemudian hitung bobot setiap jarak antar simpul.
2. Atur semua node yang telah dilalui dan atur node awal sebagai titik awal.
3. Atur semua node yang telah dilalui dan atur node awal sebagai titik.
4. Dari titik awal node, perhatikan node tetangga yang belum dilintasi dan hitung jarak dari titik awal. Jika jaraknya lebih kecil dari sebelumnya, simpan jarak baru.
5. Node yang telah dilewati tidak akan pernah diperiksa lagi.
6. Dan selanjutnya akan menemukan jarak terpendek dari setiap node yang baru disimpan

Jurnal Widya
Volume 3, Nomor 1, April 2022: halaman 59-67
https://jurnal.amikwidyaloka.ac.id/index.php/awl
jurnal@amikwidyaloka.ac.id / editor.jurnalwidya@gmail.com
P-ISSN: 2746-5411
e-ISSN: 2807-5528

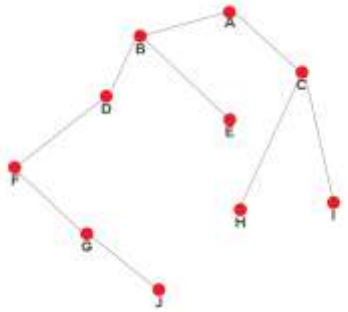
4 Hasil dan Pembahasan

Dari peta yang diambil dapat dibuat tabel yang berisi setiap tabel yang berisi setiap jarak dari setiap node.

Dari	Ke	Jarak Tempuh
A	B	350
A	C	900
B	D	1300
B	E	450
C	E	350
C	H	700
C	I	2700
D	F	400
D	G	650
D	E	650
E	C	1200
F	G	1600
G	H	2300
G	J	900
H	I	2400
H	J	600

A. Algoritma DFS

Algoritma DFS selalu mencari node terdalam untuk memastikan semua node telah dilewati. Urutan penyelesaiannya adalah ABDFGJECHI. Dengan prioritas untuk memasukkan simpul anak dari kiri. Sehingga jika digambar dengan grafik akan menjadi seperti ini :



Gambar 9. DFS

Dengan iterasi berikut:

1. Dari titik A sebagai titik awal turun ke titik turun ke titik B dan seterusnya.
2. Prioritas untuk menavigasi semua node di sisi kiri kemudian pindah ke kanan
3. Sampai Anda mencapai simpul terdalam, lalu kembali ke atas untuk menemukan simpul anak yang belum terlewatkan.
4. Sampai titik E dan ke C kemudian H dan I, dengan prioritas sisi kiri grafik terlebih dahulu.

Jarak yang dibutuhkan untuk mencapai tujuan dari titik awal dengan algoritma ini adalah **4.850 m**

Jurnal Widya
Volume 3, Nomor 1, April 2022: halaman 59-67
https://jurnal.amikwidyaloka.ac.id/index.php/awl
jurnal@amikwidyaloka.ac.id / editor.jurnalwidya@gmail.com
P-ISSN: 2746-5411
e-ISSN: 2807-5528

Pseudocode dari algoritma DFS:

```
void DFS_printPreOrder(simpul *root){
    if(root!=NULL){
        stack S;
        createEmptyStack(&S);
        push(root, &S);

        while(isStackEmpty(S) != 1){
            printf("%c ", S.top->elemen->huruf);
            simpul *node = peek(S);
            pop(&S);

            if(node->right != NULL){
                push(node->right, &S);
            }
            if(node->left != NULL){
                push(node->left, &S);
            }
        }
        printf("\n");
    }
}
```

Gambar 10. Pseudocode DFS

B. Algoritma BFS

Algoritma BFS memiliki karakteristik untuk melintasi semua node yang ada tanpa terkecuali. Semua node yang ada harus dilewati sampai akhirnya mencapai node akhir. Urutan penyelesaiannya adalah ABCDEHIFGJ. Dengan iterasi berikut:

1. Titik A sebagai lapisan 1
2. Titik B, C sebagai lapisan 2
3. Titik D, E, H, I, sebagai lapisan 3
4. Titik F sebagai lapisan 4
5. Titik G sebagai lapisan 5
6. Titik J sebagai lapisan 6
7. Dengan karakteristik dan algoritma BFS, semua node harus dilewati. Untuk mempermudah, ini dibagi per layer yang menunjukkan ketika tidak ada lagi node di layer yang belum diturunkan maka pergi ke layer berikutnya untuk menemukan akhir grafik.

Jarak yang dibutuhkan untuk mencapai tujuan dari titik awal dengan algoritma ini jauh lebih besar dibandingkan saat menggunakan algoritma DFS karena melintasi semua node untuk menemukan ujung grafik. Deskripsi matematis dari algoritma BFS adalah O (V + E), dengan V mewakili jumlah node dan E mewakili jumlah edge.

Pseudocode dari algoritma BFS:

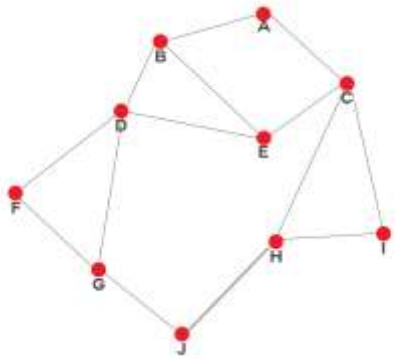
```
void BFS_printLevelOrder(simpul *root){
    if(root!=NULL){
        queue Q;
        createEmptyQueue(&Q);
        addQueue(root, &Q);

        while(isQueueEmpty(Q) != 1){
            printf("%c ", Q.first->elemen->huruf);

            if(Q.first->elemen->left != NULL){
                addQueue(Q.first->elemen->left, &Q);
            }
            if(Q.first->elemen->right != NULL){
                addQueue(Q.first->elemen->right, &Q);
            }
            delQueue(&Q);
        }
        printf("\n");
    }
}
```

Gambar 11. Pseudocode BFS

Jurnal Widya
Volume 3, Nomor 1, April 2022: halaman 59-67
https://jurnal.amikwidyaloka.ac.id/index.php/awl
jurnal@amikwidyaloka.ac.id / editor.jurnalwidya@gmail.com
P-ISSN: 2746-5411
e-ISSN: 2807-5528

C. Algoritma Dijkstra

Dengan karakteristik menemukan simpul terpendek dari titik awal hingga titik akhir, algoritma Dijkstra merupakan algoritma yang paling efektif dan tercepat dalam mencari jarak terpendek untuk mencapai tujuan akhir. Berikut penjelasannya:



Gambar 12. Dijkstra

Jarak terpendek adalah ACHJ dengan jarak **2.200 m.** Dengan iterasi berikut:

1. Titik A adalah titik awal.
2. Algoritma serakah akan membandingkan biaya jarak dari setiap node.
3. Dan akan didapatkan simpul tercepat dari semua kemungkinan simpul yang dilewati.
4. ACHJ adalah yang terpendek dari ABDFGJ, ABDGJ, ACIHJ.

Prinsip dari algoritma greedy adalah tidak mempedulikan node atau node lain yang belum tersentuh, selama algoritma ini telah menemukan titik akhir tujuan, algoritma ini akan berhenti dan tidak akan mencari lagi.

Pseudocode dari algoritma Dijkstra:

```
function Dijkstra(Graph, source):
    for each vertex v in Graph:
        dist[v] := infinity ;
        previous[v] := undefined ;
    end for
    dist[source] := 0 ;
    Q := the set of all nodes in Graph ;
    while Q is not empty:
        u := vertex in Q with smallest distance in dist[] ;
        remove u from Q ;
        if dist[u] = infinity:
            break ;
        end if
        for each neighbor v of u:

            alt := dist[u] + dist_between(u, v) ;
            if alt < dist[v]:
                dist[v] := alt ;
                previous[v] := u ;
                decrease-key v in Q;
            end if
        end for
    end while
return dist;
```

Gambar 13. Pseudocode Dijkstra

Jurnal Widya
Volume 3, Nomor 1, April 2022: halaman 59-67
https://jurnal.amikwidyaloka.ac.id/index.php/awl
jurnal@amikwidyaloka.ac.id / editor.jurnalwidya@gmail.com

P-ISSN: 2746-5411
e-ISSN: 2807-5528

5 Kesimpulan

Berdasarkan hasil perbandingan yang telah dilakukan, diperoleh kesimpulan sebagai berikut:

a. Setelah melewati tahap pengujian maka diperoleh kesimpulan bahwa Algoritma Dijkstra merupakan algoritma yang paling tepat untuk menemukan rute terpendek pada peta geografis, kemudian diikuti oleh algoritma DFS dan BFS.

b. Dalam pengujian algoritma tidak memperhatikan node yang ada, dengan kata lain tidak masalah apakah harus diurutkan seperti algoritma DFS yang mengutamakan harus pada 1 sisi graf kemudian setelah itu berpindah ke sisi graf setelahnya ..

c. Algoritma BFS kurang cocok diterapkan pada peta geografis untuk menjangkau jalur terpendek. Karena sifatnya yang menelusuri semua node terlebih dahulu sehingga membutuhkan banyak jarak dan waktu yang tidak efisien untuk mencapai titik akhir.

Referensi *(Reference)*

[1] Abba Suganda Girsang, Algortima BFS,*https://scholar.binus.ac.id/*, 2017.

[2] Aprilia, Ira. "Analisis dan Penyelesaian Permainan River Crossing Ultimate Menggunakan Algoritma BFS dan DFS." Energy-Jurnal Ilmiah Ilmu-Ilmu Teknik 6.2 (2016): 17-20.

[3] Aryono Prihandito, Proyeksi Peta, beton pertama, Yogyakarta: Kanisius, 1988.

[4] Andrew Goldberg, Microsoft Silicon Valley, penutup karpet sirkuit.

[5] Derwin Suhartono, Algoritma DFS,*https://scholar.binus.ac.id/*, 2013.

[6] Derwin Suhartono, Algoritma DFS,*https://scholar.binus.ac.id/*, 2013.

[7] Edy Irwansyah, “Sistem Informasi Geografis”, halaman 1, 2013.

[8] Fauzi, Imron. "Penggunaan algoritma dijkstra dalam pencairan rute tercepat dan rute terpendek: studi kasus pada jalan raya antara wilayah Blok M dan Kota." (2011).

[9] Purwananto, Yudhi, Diana Purwitasari, and Agung Wahyu Wibowo. "Implementasi dan analisis algoritma pencarian rute terpendek di Kota Surabaya." Jurnal penelitian dan Pengembangan TELEKOMUNIKASI 10.2 (2005): 94-101.

[10] Tobing, Fenina Adline Twince, and James Ronald Tambunan. "Analisis Perbandingan Efisiensi Algoritma Brute Force dan Divide and Conquer dalam Proses Pengurutan Angka." Ultimatics: Jurnal Teknik Informatika 12.1 (2020): 52-58.