Penerapan Algoritma Dijkstra dalam Menentukan Rute Bis Stasiun Tegalluar – Alun-Alun Bandung

Muhammad Azhar Faturahman - 13519020[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13519020@std.itb.ac.id

***Abstrak*—Pembangunan suatu daerah harus bersamaan dengan dibangunnya sarana infrakstruktur, tak terkecuali sarana transportasi. Dalam membangun sarana transportasi tentu diperlukan pemilihan rute baru yang dapat memaksimalkan potensi yang ada. Pemilihan rute yang paling mangkus dapat diselesaikan dengan menggunakan pendekatan persoalan lintasan terpendek (*Shortest Path Problem*) yang ada dalam cangkupan konsep Teori Graf. Salah satu algoritma untuk menyelesaikan persoalan lintasan terpendek adalah dengan Algoritma Dijkstra. Oleh karena itu dalam menyelesaikan persoalan untuk menentukan rute bis yang paling mangkus dapat menggunakan Algoritma Dijkstra.**

***Kata Kunci*—Teori Graf, Persoalan Lintasan Terpendek, Algoritma Dijkstra, Rute.**

I. PENDAHULUAN

Bandung Raya merupakan salah satu metropolitan yang ada di Provinsi Jawa Barat, Indonesia. Daerah Metropolitan Bandung Raya meliputi Kota Bandung, Kota Cimahi, Kabupaten Bandung, Kabupaten Bandung Barat, dan sebagian Kabupaten Sumedang. Hingga saat ini, daerah Metropolitan Bandung Raya masih terus berkembang, dan salah satu kebutuhan dalam membangun Metropolitan Bandung Raya adalah sarana dan prasarana transportasi untuk menghubungkan antar daerah [1].

Tegalluar merupakan salah satu wilayah yang sedang menjadi fokus pembangunan Metropolitan Bandung Raya. Hal ini didorong dengan pembangunan stasiun Kereta Cepat Bandung-Jakarta di wilayah tersebut [2]. Akan tetapi, jarak antara Stasiun Tegalluar dengan pusat Kota Bandung tergolong jauh, yaitu sekitar 12 Km. Oleh karena itu, untuk mempermudah akses perlu dibangun sarana Transportasi untuk menghubungkan Stasiun Tegalluar dengan pusat Kota Bandung (Alun-Alun Bandung) [3].

Salah satu sarana transportasi yang paling mudah dibangun adalah bis. Alasan pemilihan moda transportasi bis antara lain karena bisa menggunakan jalan yang sudah ada, bisa memuat banyak penumpang, dapat berhenti sesuai dengan halte dalam trayeknya, lebih ramah lingkungan, dan biaya pengadaan bis yang cenderung lebih murah dibandingkan dengan moda transportasi lain.

Dalam membangun moda transportasi bis tentu perlu ditetapkan rute yang akan dilalui oleh bis tersebut, salah satu cara untuk menyelesaikan persoalan pemilihan rute bis tersebut dapat menggunakan konsep Teori Graf. Dengan menggunakan Algoritma Dijkstra, dapat ditentukan rute bis yang paling sangkil dan juga mangkus untuk diterapkan.

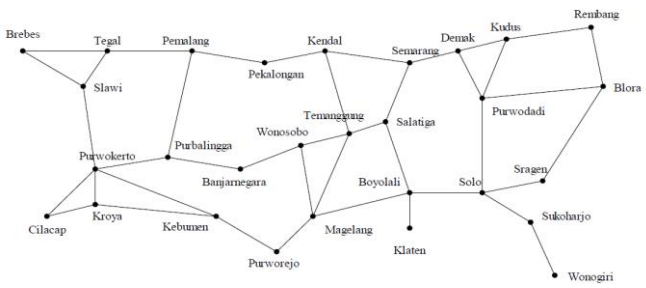
Gambar 1. Moda Transportasi Bis

Diambil dari https://bandungkunafe.com/wp-content/uploads/2017/12/bus-kota-bandung.jpg dan diakses pada 4 Desember 2020.

II. LANDASAN TEORI

A. Teori Graf

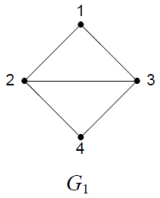
Graf digunakan untuk merepresentasikan objek-objek diskrit dan hubungan antara objek-objek tersebut. Sebagai contoh graf dapat digunakan untuk merepresentasikan peta jaringan jalan yang menghubungkan kota dalam suatu wilayah.



Gambar 2. Graf yang merepersentasikan jaringan jalan antar kota [4]

Pada Gambar 2, setiap kota direpresentasikan dengan simpul (*vertex*) dan setiap jalan direpresentasikan dengan sisi (*edge*).

Secara formal, graf didefinisikan sebagai $G = (V, E)$, yang dalam hal ini V adalah himpunan tidak kosong dari simpul-simpul (*verticles*) dan E adalah himpunan sisi (*edges*) yang menghubungkan sepasang simpul.



G_1

$$V = \{v_1, v_2, \dots, v_n\}$$
$$E = \{e_1, e_2, \dots, e_n\}$$

Graf G1 pada Gambar 3 adalah graf dengan :

$V = \{1,2,3,4\}$

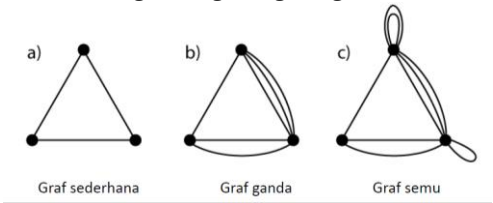
$E = \{(1,2), (1,3), (2,3), (2,4), (3,4)\}$

Gambar 3. Contoh graf [4]

B. Jenis-Jenis Graf

Berdasarkan ada tidaknya gelang atau sisi ganda pada suatu graf, maka graf digolongkan menjadi dua jenis :

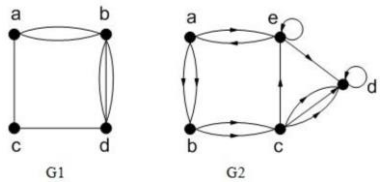
1. Graf sederhana (*simple graph*)
   Graf yang tidak mengandung gelang maupun sisi ganda.
2. Graf tak-sederhana
   Graf yang mengandung sisi ganda atau gelang, graf jenis ini dibedakan lagi menjadi :
   a. Graf ganda (*multi-graph*)
      Graf mengandung sisi ganda.
   b. Graf semu (*pseudo-graph*)
      Graf mengandung sisi gelang.



Gambar 4. Jenis graf berdasarkan ada atau tidaknya sisi ganda atau sis gelang [4]

Berdasarkan orientasi arah pada sisi, graf dibedakan atas 2 jenis :

1. Graf tak-berarah (*undirected graph*)
   Graf yang sisinya tidak mempunya orientasi arah.
2. Graf berarah (*directed graph* atau *digraph*)
   Graf yang setiap sisinya diberikan orientasi arah.



Gambar 5. Jenis graf berdasarkan orientasi arah pada sisinya, G1 *merupakan graf tak-berarah,* G2 *merupakan graf berarah* [4]

Tabel. I meringkas jenis-jenis graf berdasarkan orientasi arah dan ada tidaknya sisi ganda atau sisi gelang.

Jenis	Sisi	Sisi Ganda	Sisi Gelang
Graf sederhana	Tak-berarah	Tidak	Tidak
Graf ganda	Tak-berarah	Ya	Tidak
Graf semu	Tak-berarah	Ya	Ya
Graf berarah	Berarah	Tidak	Ya
Graf-ganda berarah	Berarah	Ya	Ya

Tabel I. Jenis-Jenis Graf [4]

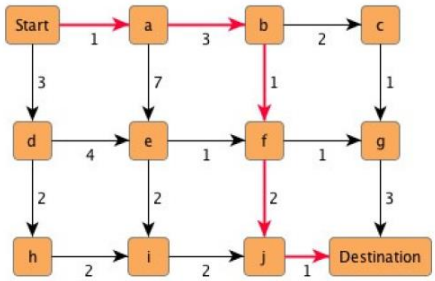
C. Terminologi Graf

1. Ketetanggaan (*Adjacent*)
   Dua buah simpul dikatakan bertetangga bila keduanya terhubung langsung.
2. Bersisian (*Incidency*)
   Untuk sembarang sisi $e = (v_i, v_j)$ dikatakan e bersisian dengan simpul v_i atau e bersisian dengan simpul v_j.
3. Simpul Terpencil (*Isolated Vertex*)
   Simpul terkecil ialah simpul yang tidak mempunya sisi yeng bersisian dengannya.
4. Graf Kosong (*Null Graph* atau *Empty Graph*)
   Graf yang himpunan sisinya merupakan himpunan kosong.
5. Derajat (*Degree*)
   Derajat suatu simpul adalah jumlah sisi yang bersisian dengan simpul tersebut.
6. Lintasan (*Path*)
   Lintasan yang panjangnya n dari simpul awal v_0 ke simpul v_n di dalam graf G ialah barisan berselang-seling simpul dan sisi-sisi yang berbentuk :
   $$v_0, e_1, v_1, e_2, v_2, \dots, v_{n-1}, v_n$$
   sedemikian sehingga $e_1 = (v_0, v_1)$, $e_1 = (v_1, v_2)$, … , $e_1 = (v_{n-1}, v_n)$ adalah sisi-sisi dari graf G.
7. Sirkuit (*Circuit*)
   Sirkuit adalah lintasan yang berawal dan berakhir pada simpul yang sama.
8. Keterhubungan (*Connected*)
   Dua buah simpul v_1 dan simpul v_2 disebut terhubung jika terdapat lintasan dari v_1 ke v_2. Graf G disebut graf terhubung (*connected graph*) jika untuk setiap pasang simpul v_i dan v_j dalam himpunan V terdapat lintasan dari v_i ke v_j.
9. Upagraf (*Subgraph*) dan Komplemen Upagraf
   Misalkan $G = (V, E)$ adalah sebuah graf. $G_1 = (V_1, E_1)$ adalah upagraf dari G jika $V_1 \subseteq V$ dan $E_1 \subseteq E$.
   Komplemen dari upagraf G_1 terhadap graf G adalah graf $G_2 = (V_2, E_2)$ sedemikian sehingga $E_2 = E - E_1$ dan V_2 adalah himpunan simpul yang anggota-anggota E_2 bersisian dengannya.
10. Upagraf Merentang (*Spanning Subgraph*)
    Upagraf $G_1 = (V_1, E_1)$ dari $G = (V, E)$ dikatakan upagraf rentang jika $V_1 = V$.
11. Cut-Set
    Cut-set dari graf terhubung G adalah himpunan sisi yang bila dibuang dari G menyebabkan G tidak terhubung.
12. Graf Berbobot (*Weighted Graph*)
    Graf berbobot adalah graf yang setiap sisinya diberi sebuah harga (bobot).

D. Persoalan Lintasan Terpendek

Persoalan lintasan terpendek (*shortest path problem*)

merupakan salah satu masalah klasik yang ada dalam Teori Graf. Pada persoalan ini diberikan sebuah graf berbobot dengan *n* buah simpul, lalu akan ditentukan lintasan terpendek atau lintasan yang akan memberikan nilai paling optimum untuk bergerak dari suatu simpul ke simpul lainnya.



Gambar 6. Ilustrasi Persoalan Lintasan Terpendek [5]

Persoalan lintasan terpendek mempunyai banyak kegunaan dalam kehidupan sehari-hari seperti dalam menentukan rute terpendek dalam peta untuk bergerak dari suatu titik ke titik lain. [4]

E. Algoritma Greedy

Algoritma *greedy* merupakan suatu metode untuk menyelesaikan persoalan optimasi berupa mencari nilai maksimum (*maximization*) atau mencari nilai minimum (*minimization*).

Pada prinsipnya, algoritma *greedy* akan membentuk solusi langkah per langkah dengan mengambil solusi terbaik pada langkah tersebut (*"take what you can get now!"*). Pada setiap langkah, terdapat banyak pilihan yang harus dievaluasi. Oleh karena itu, pada setiap langkah harus dibuat keputusan yang terbaik dalam menentukan pilihan.

Di setiap langkahnya, akan dicari nilai optimum local (*local optimum*) dengan harapan langkah sisanya mengarah ke solusi optimum global (*global optimum*).

Algoritma *greedy* melibatkan pencarian sebuah himpunan bagian S, dari himpunan kandidat C, yang dalam hal ini, S harus memenuhi beberapa kriteria yang ditentukan, yaitu menyatakan suatu solusi dan S dioptimisasi oleh fungsi objektif.

Penggunaan algoritma *greedy* tidak selalu memberikan optimum global, tetapi dapat memberikan optimum local atau *pseudo-optimum*. Jika jawaban terbaik mutlak tidak diperlukan maka algoritma *greedy* sering berguna untuk menghasilkan solusi hampiran. Bila algoritma *greedy* optimum, maka keoptimalannya itu dapat dibuktikan secara matematis.

Salah satu algoritma *greedy* yang banyak digunakan untuk menyelesaikan persoalan lintasan terpendek adalah menggunakan Algoritma Dijkstra. [5]

F. Algoritma Dijkstra

Algoritma Dijkstra merupakan algoritma yang optimal untuk menentukan lintasan terpendek. Pada setiap langkah, ambil sisi yang berbobot minimum yang menghubungkan sebuah simpul yang sudah terpilih dengan sebuah simpul lain yang belum terpilih. Lintasan dari simpul asal ke simpul yang baru haruslah merupakan lintasan yang terpendek diantara semua lintasannya ke simpul-simpul yang belum terpilih.

Berikut merupakan *pseudocode* dari Algoritma Dijkstra :

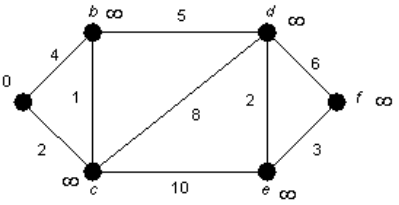
```
procedure Dijkstra (input G: weighted_graph, input
a: initial_vertex)

DEKLARASI
 S : himpunan simpul solusi
 L : array[1..n] of real {L[z] berisi Panjang
lintasan terpendek dari a ke z}

ALGORITMA
for i ← 1 to n
   L(vi) ← ∞
end for
L(a) ← 0
S ← {}
while z ∉ S do
   u ← simpul yang bukan di dalam S dan memiliki
L(u) minimum
   S ← S ∪ {u}
   for semua simpul v yang tidak terdapat didalam S
      if L(u) + G(u,v) < L(v) then
         L(v) ← L(u) + G(u,v)
      end for
   end while
```

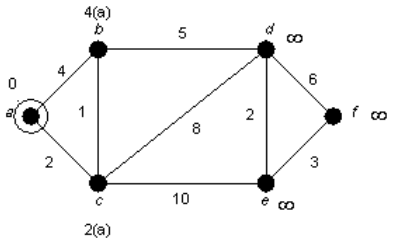
Contoh penggunaan Algoritma Dijkstra :

1. Inisialisasi semua nilai simpul dengan infinite. Simpul awal (a) inisialisasi dengan nilai 0.



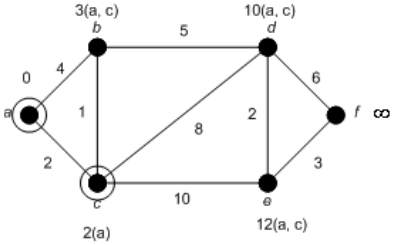
Gambar 7. Langkah 1 [5]

2. Ambil simpul a dengan bobot minimum, lalu hitung bobot simpul di sekitar simpul a.



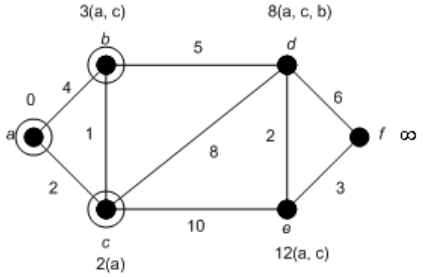
Gambar 8. Langkah 2 [5]

3. Ambil simpul c dengan bobot minimum, lalu hitung bobot simpul di sekitar c yang belum dipilih. Perhartikan simpul b, nilai sebelumnya adalah 4 dengan lintasan melalui a, setelah dihitung kembali nilainya menjadi 3 dengan lintasan melalui a,c.



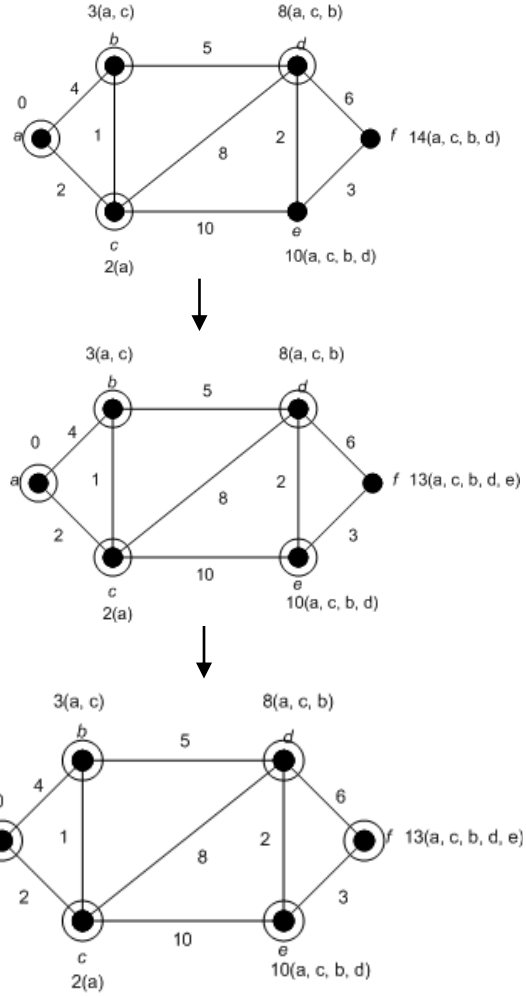
Gambar 9. Langkah 3 [5]

4. Ambil simpul dengan bobot minimum dalam hal ini simpul b, lalu hitung nilai simpul disekitar simpul b yang belum dipilih dan evaluasi nilai yang sebelumnya sudah ada seperti pada langkah 3.



Gambar 10. Langkah 4 [5]

5. Ulangi langkah 2,3,4 hingga simpul tujuan terpilih.



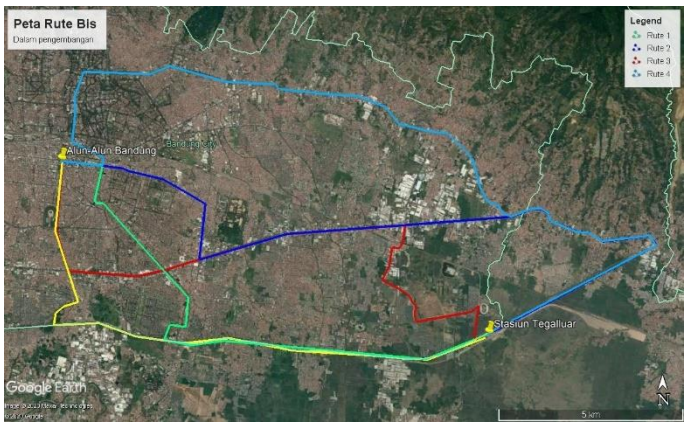
Gambar 11. Langkah 5, 6, dan 7 [5]

6. Setelah simpul tujuan terpilih, maka lintasan terpendek berhasil didapatkan. [5]

III. ANALISIS PERSOALAN

A. Pemodelan Rute Bis

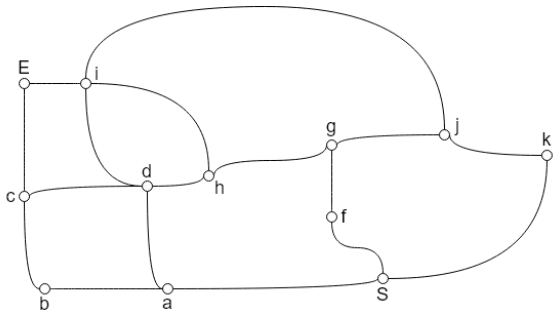
Dalam makalah ini, penulis memilih beberapa rute yang mungkin dapat dilalui oleh bis berdasarkan jenis jalan yang tersedia. Dengan titik awal dari Stasiun Tegalluar dan titik akhir berada di Alun-Alun Bandung.



Gambar 12. Rute bis yang dipilih penulis

(Sumber : Dokumen Pribadi / Google Earth Pro)

Untuk mempermudah penyelesaian persoalan, maka dilakukan pemodelan dengan mengubah rute pada Gambar. 12 menjadi sebuah graf, dengan setiap simpul merepresentasikan persimpangan dan setiap sisi yang merepresentasi ruas jalan.



Gambar 13. Hasil pemodelan peta rute

(Sumber : Dokumen Pribadi)

Pada Gambar. 13, simpul S merepresentasikan Stasiun Tegalluar dan Simpul E merepresentasikan Alun-Alun Bandung.

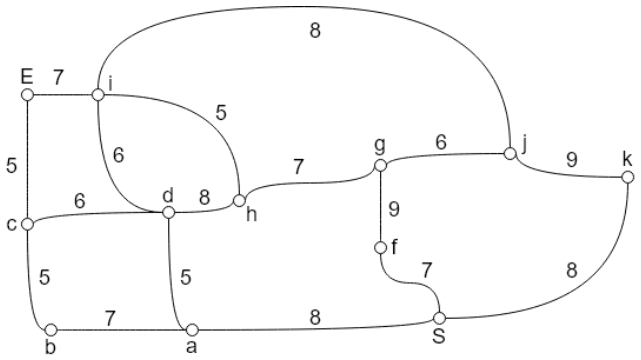
Sisi	**Ruas Jalan yang direpresentasikan**
(S,a)	Tol Purbalenyi (Ruas Gede Bage – Buah Batu)
(S,f)	Stasiun Tegalluar - Gede Bage
(S,k)	Tol Purbalenyi (Ruas Gede Bage - Cileunyi)
(a,b)	Tol Purbalenyi (Ruas Buah Batu – Moh. Toha)
(a,d)	Terusan Buah Batu – Soekarno Hatta
(b,c)	Moh. Toha – Seokarno Hatta
(c,d)	Moh. Toha – Buah Batu
(c,E)	Moh. Toha – Alun-Alun Bandung
(d,h)	Buah Batu – Kiaracondong
(d,i)	Buah Batu – Karapitan – Asia Afrika
(f,g)	Gede Bage – Soekarno Hatta
(g,h)	Gede Bage – Kiaracondong
(g,j)	Gede Bage – Cibiru
(h,i)	Kiaracondong – Gatot Subroto – Asia Afrika
(i,E)	Asia Afrika – Alun-Alun Bandung
(j,i)	Cibiru – A.H. Natsution – Surapati – Merdeka – Asia Afrika
(j,k)	Cibiru - Cileunyi

Tabel II. Representasi Ruas Jalan oleh Sisi pada Graf

Graf yang dimodelkan pada Gambar. 13 dan Tabel 2. masih berupa graf tak-berbobot, oleh karena itu diperlukan pembobotan berdasarkan parameter tertentu. Dalam hal ini penulis memutuskan untuk menggunakan 3 parameter, yaitu Jarak, Kecepatan, dan Aksesbilitas. Setiap parameter mempunyai nilai dari 1-5. Parameter jarak dengan nilai 1 berarti sangat dekat dan nilai 5 berarti sangat jauh (1 satuan nilai ≈ 4 Km), parameter kecepatan dengan nilai 1 berarti jalur tidak padat kendaraan dan nilai 5 berarti sering terjadi kemacetan, parameter aksesbilitas dengan nilai 1 menyatakan jalan melalui tempat yang dapat diakses dan mencangkup banyak penduduk sedangkan nilai 5 menyatakan jalan tidak dapat diakses penduduk.

Sisi	Jarak	Kecepatan	Aksesbilitas	Total
(S,a)	2	1	5	8
(S,f)	1	3	3	7
(S,k)	2	1	5	8
(a,b)	1	1	5	7
(a,d)	1	2	2	5
(b,c)	1	2	2	5
(c,d)	1	3	2	6
(c,E)	1	3	1	5
(d,h)	1	3	4	8
(d,i)	1	4	1	6
(f,g)	1	4	4	9
(g,h)	2	4	1	7
(g,j)	1	2	3	6
(h,i)	1	3	1	5
(i,E)	1	3	3	7
(j,i)	4	3	1	8
(j,k)	2	4	3	9

Tabel III. Hasil pembobotan sisi graf berdasarkan parameter tertentu



Gambar 14. Graf Berbobot hasil pemodelan ruas jalan

(Sumber : Dokumen Pribadi)

B. Algoritma Dijkstra dalam Menentukan Rute Paling Mangkus

Dari hasil pembobotan yang diberikan pada Tabel. III dan ditampilkan pada graf di Gambar. 14, akan dilakukan pencarian rute yang paling efisien menggunakan Algoritma Dijkstra. Pencarian rute terpendek dilakukan dengan simpul S sebagai simpul awal dan simpul E sebagai simpul tujuan.

1. Inisialisasi

Simpul	Nilai	Lintasan	Status
S	0	-	-
a	∞	-	-
b	∞	-	-
c	∞	-	-
d	∞	-	-
f	∞	-	-
g	∞	-	-
h	∞	-	-
i	∞	-	-
j	∞	-	-
k	∞	-	-
E	∞	-	-

Tabel IV. Inisialisasi Dijkstra

2. Langkah ke-1, ambil simpul S, lalu hitung nilai simpul a,f, dan k.

Simpul	Nilai	Lintasan	Status
S	0	S	✓
a	8	S	-
b	∞	-	-
c	∞	-	-
d	∞	-	-
f	7	S	-
g	∞	-	-
h	∞	-	-
i	∞	-	-
j	∞	-	-
k	8	S	-
E	∞	-	-

Tabel V. Langkah 1

3. Langkah ke-2, ambil simpul f, lalu hitung nilai simpul g.

Simpul	Nilai	Lintasan	Status
S	0	S	✓
a	8	S	-
b	∞	-	-
c	∞	-	-
d	∞	-	-
f	7	S	✓
g	16	S - f	-
h	∞	-	-
i	∞	-	-
j	∞	-	-
k	8	S	-
E	∞	-	-

Tabel VI. Langkah 2

4. Langkah ke-3, ambil simpul a, lalu hitung nilai simpul b dan d.

Simpul	Nilai	Lintasan	Status
S	0	S	✓
a	8	S	✓
b	15	S - a	-
c	∞	-	-
d	13	S - a	-
f	7	S	✓
g	16	S - f	-
h	∞	-	-
i	∞	-	-
j	∞	-	-
k	8	S	-
E	∞	-	-

Tabel VII. Langkah ke-3

5. Langkah ke-4 dan ke-5, ambil simpul k dan d, lalu hitung nilai simpul j, c, h, dan i.

Simpul	Nilai	Lintasan	Status
S	0	S	✓
a	8	S	✓
b	15	S - a	-
c	19	S - a - d	-
d	13	S - a	✓
f	7	S	✓
g	16	S - f	-
h	21	S - a - d	-
i	19	S - a - d	-
j	17	S - k	-
k	8	S	✓
E	∞	-	-

Tabel VIII. Langkah 4 dan 5

6. Langkah ke-6 dan ke-7, ambil simpul b dan g, tidak ada perubahan karena nilai lintasan yang melalui b atau g lebih besar dibandingkan dengan lintasan lainnya.

Simpul	Nilai	Lintasan	Status
S	0	S	✓
a	8	S	✓
b	15	S - a	✓
c	19	S - a - d	-
d	13	S - a	✓
f	7	S	✓
g	16	S - f	✓
h	21	S - a - d	-
i	19	S - a - d	-
j	17	S - k	-
k	8	S	✓
E	∞	-	-

7. Langkah ke-8 dan ke-9, ambil simpul j dan c, tidak ada perubah pada simpul g dan i karena nilai yang melalui lintasan tersebut lebih besar daripada melalui lintasan lain, kemudian hitung nilai simpul E.

Simpul	Nilai	Lintasan	Status
S	0	S	✓
a	8	S	✓
b	15	S - a	✓
c	19	S - a - d	✓
d	13	S - a	✓
f	7	S	✓
g	16	S - f	✓
h	21	S - a - d	-
i	19	S - a - d	-
j	17	S - k	✓
k	8	S	✓
E	24	S - a - d - c	-

Tabel IX. Langkah 8 dan 9

8. Langkah ke-10 dan ke-11, ambil simpul i dan h, tidak ada perubahan pada simpul manapun karena nilai yang melalui lintasan tersebut lebih besar daripada yang melalui lintasan lain yang sudah dihitung.

Simpul	Nilai	Lintasan	Status
S	0	S	✓
a	8	S	✓
b	15	S - a	✓
c	19	S - a - d	✓
d	13	S - a	✓
f	7	S	✓
g	16	S - f	✓
h	21	S - a - d	✓
i	19	S - a - d	✓
j	17	S - k	✓
k	8	S	✓
E	24	S - a - d - c	-

Tabel X. Langkah 10 dan 11

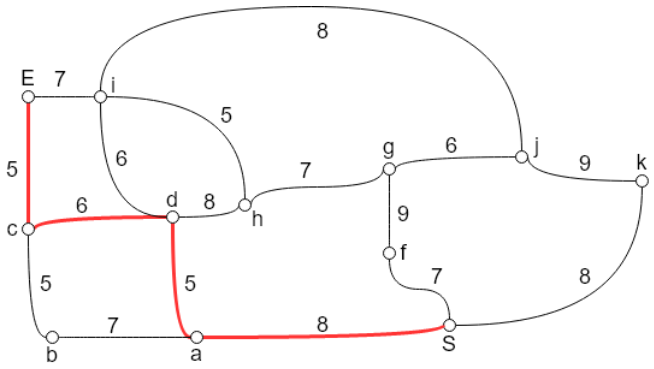
9. Langkah ke-12, ambil simpul E. Karena simpul E adalah simpul tujuan dan tidak ada lagi simpul-simpul yang perlu dianalisis, maka algoritma selesai sampai disini dan berhasil ditemukan rute paling efisien.

Simpul	Nilai	Lintasan	Status
S	0	S	✓
a	8	S	✓
b	15	S - a	✓
c	19	S - a - d	✓
d	13	S - a	✓
f	7	S	✓

g	16	S - f	✓
h	21	S - a - d	✓
i	19	S - a - d	✓
j	17	S - k	✓
k	8	S	✓
E	24	S - a - d - c	✓

Tabel XI. Langkah 12

Dari Tabel. XI dapat ditentukan lintasan yang paling mangkus dari simpul S ke simpul E adalah dengan melalui simpul a, simpul d, dan simpul c.



Gambar 15. Hasil pencarian rute paling mangkus dengan Algoritma Dijkstra

(Sumber : Dokumen Pribadi)

Dalam artiannya di dunia nyata, maka rute bis yang paling mangkus adalah :
Stasiun Tegalluar → Tol Purbalenyi → Jl. Terusan Buah Batu → Jl. Soekarno Hatta → Jl. Moh. Toha → Alun-Alun Bandung Rute ini akan memiliki jarak tempuh sejauh 17,39 Km dan waktu tempuh sekitar 35 menit dengan asumsi kecepatan rata-rata bis disekitar 30 Km/Jam.

IV. Analisis Solusi

Solusi yang berhasil diberikan pada Bagian III masih memiliki banyak kekurangan, antara lain :

1. Solusi yang diberikan hanya dapat diterapkan untuk rute Stasiun Tegalluar – Alun-Alun Bandung. Hal ini dikarenakan karena pemilihan rute pada Gambar. 12 merupakan murni pemilihan penulis tanpa mempertimbangkan hal-hal lain seperti mungkin saja jalan yang menjadi pilihan penulis hanya bisa dilalui satu arah saja, selain itu kondisi ruas jalan hanya penulis kira-kira dan didapatkan melalui aplikasi Google Earth Pro.
2. Pemilihan parameter dan penentuan nilai parameter masih yang didasarkan pengalaman penulis, tanpa mempertimbangkan fakta-fakta yang ada di lapangan. Oleh karena itu, jika ingin diterapkan perlu adanya pengkajian ulang untuk menganalisis kondisi lapangan sesungguhnya yang tentu akan berpengaruh pada nilai parameter. Selain itu penulis merasa bahwa parameter yang digunakan masih terlalu sedikit, sehingga perlu ditambah parameter-parameter lain untuk menambah keakuratan hasil pemilihan rute dengan Algoritma Dijkstra.

Adapun kelebihan yang dapat diberikan adalah pada analisis persoalan di Bagian III, pemilihan algoritma dengan menggunakan Algoritma Dijkstra sudah cukup tepat, karena dapat memberikan solusi yang optimum. Selain itu pemilihan rute bis pada Gambar. 12 sudah cukup baik mengingat rute bis yang cenderung melalui jalan besar dan tidak banyak berbelok-belok.

V. Kesimpulan

Teori graf dapat digunakan untuk merepresentasikan peta jaringan jalan dalam suatu kota, dalam hal ini suatu simpul dapat merepresentasikan sebuah persimpangan dan sisi dapat merepresentasikan sebuah jalan. Representasi jaringan jalan menjadi sebuah graf dapat mempermudah dalam menyelesaikan suatu persoalan seperti dalam menentukan rute bis baru dari suatu tempat ke tempat lain.

Dalam makalah ini, penulis mengambil suatu kasus yaitu membangun rute bis baru dari Stasiun Tegalluar menuju Alun-Alun Bandung. Dengan menggunakan Algoritma Dijkstra, didapatkan hasil suatu peta rute bis baru yang paling mangkus berdasarkan pertimbangan parameter jarak, waktu, dan aksesbilitas. Rute yang didapat adalah : Stasiun Tegalluar → Tol Purbalenyi → Jl. Terusan Buah Batu → Jl. Soekarno Hatta → Jl. Moh. Toha → Alun-Alun Bandung. Rute tersebut memiliki jarak tempuh sejauh 17,39 Km dengan perkiraan waktu tempuh sekitar 35 menit.

VII. Ucapan Terima Kasih

Segala puji dan syukur penulis panjatkan kepada kehadirat Allah SWT, karena berkat rahmat, hidayah dan karunia-Nya maka penulis dapat menyelesaikan makalah ini tepat pada waktunya.

Penulis juga mengucapkan terimakasih sebesar-besarnya kepada semua pihak yang telah turut membantu penulis dalam menyelesaikan makalah ini. Terimakasih banyak kepada Ibu Nur Ulfa Maulidevi selaku dosen pengampu mata kuliah IF2120 Matematika Diskrit kelas K4, dan juga kepada seluruh tim pengajar mata kuliah IF2120 yang telah banyak memberikan ilmunya kepada penulis hingga penulis mampu menyelesaikan makalah ini.

Terimakasih juga kepada seluruh teman-teman penulis yang selalu memberikan dorongan motivasi dan memberikan saran-saran kepada penulis dalam mengerjakan tugas makalah ini.

Referensi

[1] https://humas.bandung.go.id/layanan/pemkot-dorong-pembangunan-metropolitan-bandung-raya diakses pada 4 Desember 2020.
[2] https://www.kompas.com/tren/read/2019/08/31/080732165/mengenal-tegalluar-walini-dan-rebana-calon-ibu-kota-baru-jawa-barat?page=all diakses pada 4 Desember 2020.
[3] http://humas.jabarprov.go.id/gubernur-jabar-dorong-percepatan-pembangunan-akses-penghubung-kereta-cepat-jakarta-bandung/3848 diakses pada 4 Desember 2020.
[4] R. Munir, "Diktat Kuliah Matematika Diskrit ITB (IF2120)", 2020, diakses dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/matdis20-21.htm pada 5 Desember 2020.
[5] R. Munir, "Diktat Kuliah Strategi Algoritma ITB (IF2211)", 2019, diakses dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/stmik.htm pada 5 Desember 2020.

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 Desember 2020

Muhammad Azhar Faturahman
13519020