Aplikasi Graf dalam Pencarian Rute Optimal pada Permainan Kurir Mobita

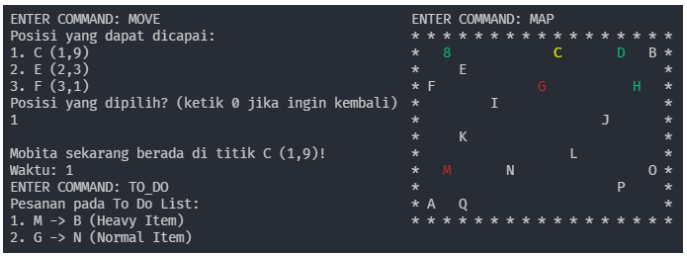
Muhammad Fikri Ranjabi - 13520002[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13520002@std.stei.itb.ac.id

***Abstrak*—Kurir Mobita merupakan sebuah permainan berbasis CLI (*command-line interface)* tentang pengantaran barang. Permainan ini memiliki sebuah karakter fiksi bernama Mobita dengan tujuan akhir untuk mengantarkan semua barang yang ada ke tempat tujuannya masing-masing sesuai dengan lokasi pada peta. Lokasi awal Mobita berada di *headquarters* dan akan berakhir di *headquarters* setelah semua barang berhasil diantar. Setiap barang memiliki lokasi antar dan lokasi ambil. Dengan aplikasi graf, dapat dicari rute perjalanan optimal yang dapat dipilih oleh Mobita agar waktu pengantaran barang menjadi efisien dan tidak ada jalur yang sama yang dilalui sebanyak dua kali.**

***Kata Kunci*—Graf, Pohon Merentang Minimum, Algoritma *Prim*, Rute Perjalanan, Sirkuit *Hamilton*.**

I. PENDAHULUAN

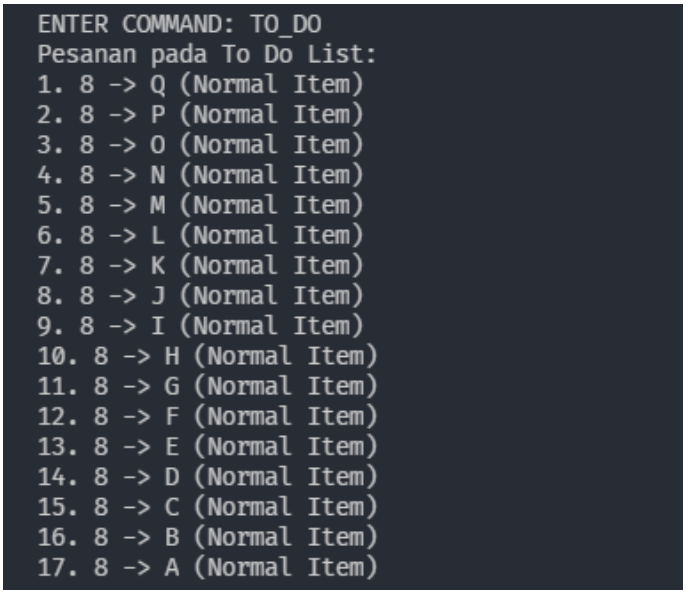
Kurir Mobita merupakan sebuah permainan berbasis CLI (*command-line interface)* tentang pengantaran barang. Permainan ini adalah hasil dari tugas besar mata kuliah IF2110 Algoritma dan Struktur Data dan dibuat menggunakan bahasa C dengan memanfaatkan berbagai struktur data abstrak seperti point, list statis, list dinamis, linked list, queue, matriks, stack, dan mesin kata.



Gambar 1. Tampilan Permainan Kurir Mobita

Permainan ini diawali kisah latar belakang yaitu sebuah karakter fiksi bernama Mobita yang ingin membantu orang tuanya untuk mendapatkan penghasilan dengan menjadi seorang kurir pengantar barang. Mobita dapat memiliki kemampuan untuk melacak pesanan, navigasi peta, serta mengambil dan menurunkan barang yang ia antar. Mobita akan mendapatkan uang setiap pesanan yang berhasil diantar.

Tujuan dari permainan ini adalah untuk mengambil pesanan dan mengantarkan semua pesanan ke lokasi yang bersesuaian. Saat permainan baru pertama kali dimulai, Mobita akan berada di posisi *headquarters* (angka 8 pada peta) dan menerima beberapa daftar pesanan yang akan diantar. Gambar 2 menampilkan daftar pesanan yang harus diantar oleh Mobita. Daftar pesanan, model peta, dan rute perjalanan merupakan konfigurasi yang dapat disesuaikan oleh pemain. Setiap pesanan memiliki informasi berupa waktu pesanan masuk, lokasi tempat ambil, lokasi tempat antar, jenis pesanan, dan waktu hangus bila jenis pesanan adalah *perishable item*.



Gambar 2. Daftar Pesanan yang Harus Diantar

Kemudian Mobita akan bergerak menuju lokasi *pick up* pesanan untuk mengambil pesanan yang ada pada daftar *to do list* pada gambar 2, setelah pesanan di-*pick up* maka pesanan akan masuk ke dalam tas Mobita. Gambar 3 merupakan tampilan ketika Mobita berhasil melakukan *pick up* item pada lokasi yang sudah sesuai. Dalam konfigurasi ini, pemain sebelumnya telah membeli *gadget* bernama Senter Pembesar sebanyak empat buah sehingga kapasitas maksimum tas Mobita adalah 24 item dari yang sebelumnya hanya tiga item.

Setelah berhasil melakukan *pick* up, Mobita akan bergerak menuju lokasi *drop off* untuk menaruh pesanan yang diantarkan. Pada gambar 4 dapat dilihat pesan yang ditampilkan ketika pesanan sudah berhasil diantarkan. Mobita juga mendapatkan tambahan uang sebanyak 200.

```
ENTER COMMAND: PICK_UP
Pesanan berupa Normal Item berhasil diambil!
Tujuan Pesanan: Q
```

Gambar 3. Pesanan Berhasil Diambil

```
Mobita sekarang berada di titik Q (10,3)!
Waktu: 6
ENTER COMMAND: DROP_OFF
TIPE PESANAN N
Pesanan Normal Item berhasil diantarkan
Uang yang didapatkan: 200
```

Gambar 4. Pesanan Berhasil Diantarkan

Jika semua pesanan sudah berhasil diantarkan dan Mobita kembali ke posisi *headquarters*, maka akan ada pesan yang menampilkan bahwa permainan sudah selesai beserta keterangan jumlah barang yang berhasil diantar dan total waktu yang dilampaui.

II. LANDASAN TEORI

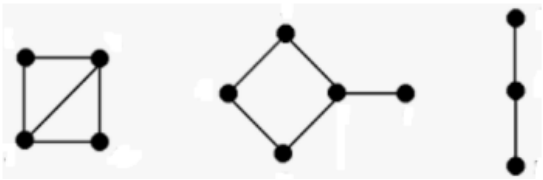
A. Graf

Graf didefinisikan sebagai pasangan simpul dan sisi. Misal sebuah graf G = (V, E) dengan V merupakan himpunan tidak kosong dari simpul-simpul (*vertices)* dan E merupakan himpunan sisi (*edges*) yang menghubungkan sepasang simpul, yaitu:

$$V = \{ V_1, V_2, V_3, \dots, V_n \}$$
$$E = \{ e_1, e_2, e_3, \dots, e_n \}$$

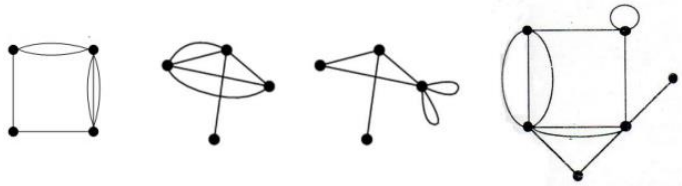
B. Jenis-Jenis Graf

1. Graf Sederhana
   Graf yang tidak mengandung gelang maupun sisi ganda.



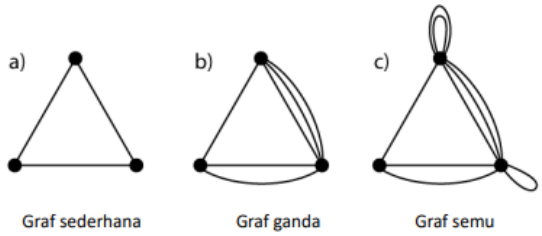
Gambar 5. Graf Sederhana

2. Graf Tak Sederhana
   Graf yang mengandung sisi ganda atau gelang.



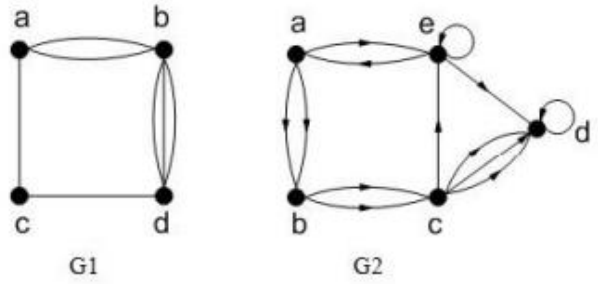
Gambar 6. Graf Tak Sederhana

Graf tak sederhana dibedakan lagi menjadi dua jenis yaitu graf ganda dan graf semu. Graf ganda adalah graf yang mengandung sisi ganda, sedangkan graf semu adalah graf yang mengandung sisi gelang.



Gambar 7. Graf Sederhana, Graf Ganda, dan Graf Semu

Berdasarkan orientasi arah pada sisi, graf dibedakan lagi menjadi dua jenis yaitu graf tak berarah dan graf berarah. Graf tak berarah sisinya tidak mempunyai orientasi arah, sedangkan pada graf berarah sisinya memiliki orientasi arah.

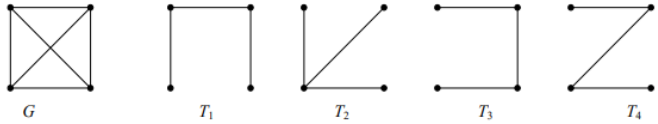


Gambar 8. Graf Tak Berarah dan Graf Berarah

B. Terminologi pada Graf

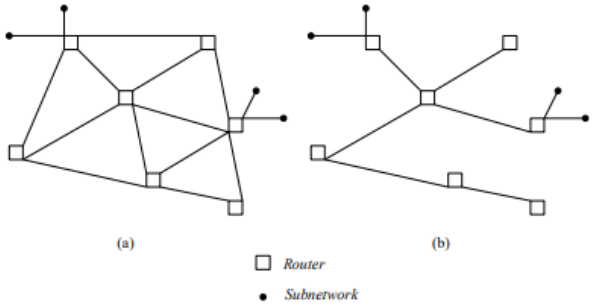
1. Ketetangaan (*Adjacent*)
   Dua buah simpul pada graf dikatakan bertetangga apabila keduanya terhubung langsung.
2. Beririsan (*Incidency*)
   Sebuah sisi graf beririsan terhadap simpul yang terhubung dengan sisi tersebut.
3. Simpul Terpencil (*Isolated Vertex*)
   Simpul pada graf yang tidak mempunyai sisi yang beririsan.
4. Graf Kosong (*Null Graph*)
   Graf yang himpunan sisinya merupakan himpunan kosong.
5. Derajat (*Degree*)
   Derajat pada simpul menyatakan jumlah sisi yang bersisian dengan simpul tersebut.
6. Lintasan (*Path*)
   Barisan berselang-seling simpul-simpul dan sisi-sisi sedemikian yang saling terhubung.
7. Siklus (*Cycle*) atau Sirkuit (*Circuit*)
   Lintasan yang berawalan dan berakhiran pada simpul yang sama.
8. Keterhubungan (*Connected*)
   Dua buah simpul dikatakan terhubunga jika terdapat lintasan yang menghubungkan dua simpul tersebut. Sebuah graf dikatakan terhubung jika untuk setiap pasang simpulnya terdapat lintasan yang menghubungkan simpul tersebut.
9. Upagraf (*Subgraph*)
   Upagraf adalah graf yang merupakan bagian dari graf yang simpul/sisinya lebih lengkap.
10. Upagraf Merentang (*Spanning Subgraph*)
    Sebuah upagraf yang memiliki semua simpul dari graf G.
11. Graf Berbobot
    Graf yang setiap sisinya memiliki sebuah harga (bobot).

C. Pohon Merentang



Gambar 9. Pohon Merentang

Pohon memiliki definisi sebuah graf tak berarah yang tidak mengandung sirkuit. Pohon merentang (*spanning tree*) dari graf terhubung adalah upagraf dari pohon yang juga merentang dan diperoleh dengan memutus sirkuit di dalam graf. Setiap graf memiliki paling sedikit satu buah pohon merentang.



Gambar 10. Pohon Merentang pada Jaringan Komputer dan Multicast

Pohon merentang memiliki banyak aplikasi dalam dunia nyata. Beberapa di antaranya yaitu untuk menggambarkan jumlah jumlah ruas minimum yang menghubungkan semua kota pada jalan sehingga setiap kota tetap terhubung satu sama lain dan perutean pesan pada jaringan komputer.

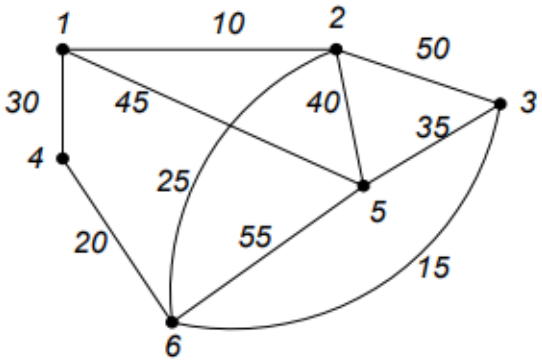
Graf terhubung berbobot mungkin mempunyai lebih dari 1 pohon merentang. Oleh karena itu, pohon merentang yang memiliki bobot minimum dinamakan pohon merentang minimum (*minimum spanning tree*).

D. Algoritma Prim

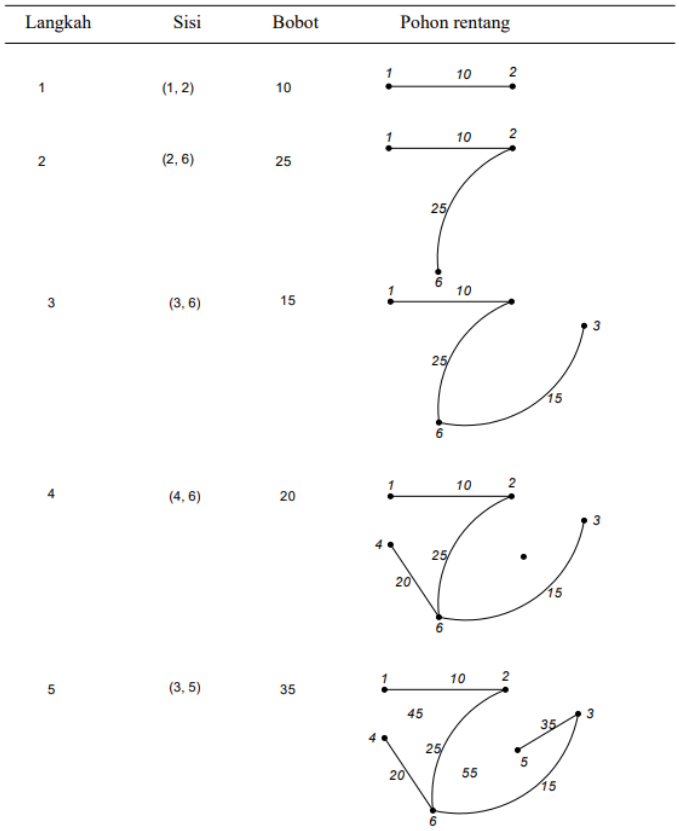
Algoritma *Prim* merupakan algoritma yang digunakan untuk mencari pohon merentang minimum dari suatu graf berbobot terhubung. Perlu dicatat bahwa pohon merentang yang dihasilkan tidak selalu unik meskipun bobotnya tetap sama. Hal ini terjadi karena terdapat beberapa sisi yang dipilih dengan bobot yang sama. Berikut adalah langkah-langkah algoritma *Prim*:

1. Ambil sisi dari graf G yang memiliki bobot minimum, kemudian masukkan ke dalam T.
2. Pilih sisi (u, v) yang mempunyai bobot minimum dan bersisian dengan simpul di T, tetapi (u, v) tidak membentuk sirkuit di T. Masukkan (u, v) ke dalam T.
3. Ulangi langkah 2 sebanyak n-2 kali.

Misal terdapat sebuah graf berbobot pada gambar 11 dan akan dicari pohon merentang minimum dari graf tersebut.

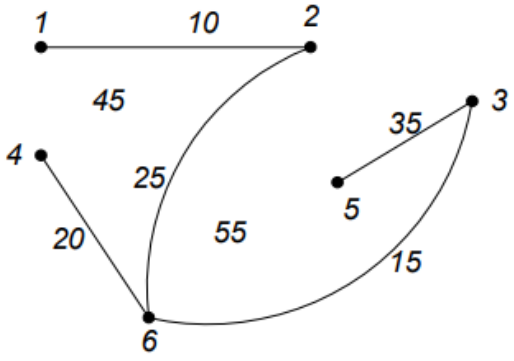


Gambar 11. Graf Berbobot



Gambar 12. Langkah dalam Mencari Pohon Merentang Minimum

Kemudian diterapkan algoritma *Prim* seperti pada gambar 12. Sehingga dihasilkan pohon merentang minimum dengan bobot total 105 pada gambar 13.

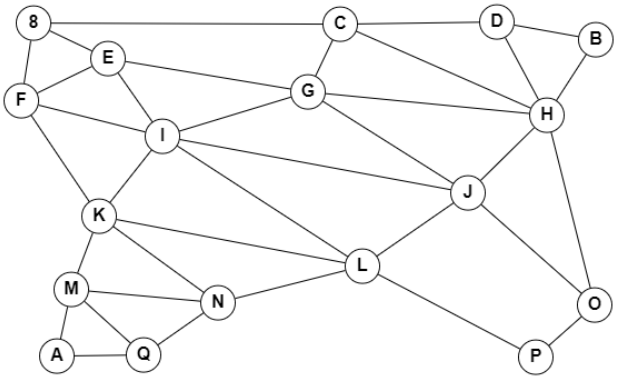


Gambar 13. Pohon Merentang Minimum

III. APLIKASI GRAF DALAM PENCARIAN RUTE YANG OPTIMAL

Pencarian rute yang optimal dalam permainan Mobita dapat memanfaatkan teori yang terdapat pada graf yaitu Sirkuit *Hamilton*. Tujuan yang ingin dicapai adalah untuk mencari rute yang optimal sehingga di dalam permainan, Mobita dapat mengirim barang ke semua titik pengantaran tanpa melalui jalan yang sama dan kembali ke titik asal yaitu *headquarters*.

Gambar 14 menunjukkan peta rute perjalanan yang harus dilalui oleh Mobita dalam bentuk graf terhubung. Peta ini direpresentasikan ke dalam bentuk *adjacency matrix* yang terdapat pada gambar 15 untuk dimuat ke dalam konfigurasi permainan. Angka delapan melambangkan lokasi *Headquarters* sebagai lokasi awal Mobita, pada lokasi ini semua pesanan Mobita masuk ke dalam tas.



Gambar 14. Peta rute perjalanan Mobita dalam bentuk graf

```
0001011000000000000
0000000000000010001
0000100010000000000
1000100110000000000
0011000010000000000
1000001101000000000
1000010001010000000
0001010011100000000
0011100100100001000
0000011100111000000
0000000111001001000
0000010010011100000
0000000011100100100
0000010010011100000
0000000001110010100
0100000000010010010
0000000000011100001
0000000010100000100
0000000000000100100
0100000000000011000
```

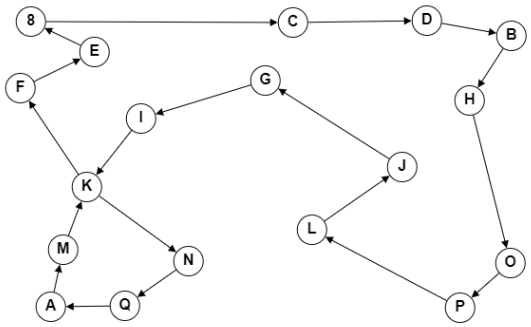
Gambar 15. Representasi graf peta dalam matriks keterhubungan

Setiap huruf pada peta merupakan lokasi yang dapat dikunjungi oleh Mobita untuk mengantarkan barang. Mobita memiliki jumlah pesanan sebanyak jumlah lokasi yaitu 17 pesanan dengan satu pesanan di setiap lokasi. Setiap perpindahan dari satu tempat ke tempat lainnya akan menghabiskan sebanyak satu satuan waktu.

Graf ini memiliki karakteristik graf berbobot terhubung dengan bobot setiap sisi adalah satu. Hal ini disebabkan setiap perpindahan lokasi menambah satu satuan waktu terhadap total waktu yang dimiliki oleh Mobita sehingga jarak setiap lokasi diasumsikan sama. Kemudian kita dapat menerapkan teori sirkuit *Hamilton* ke dalam graf ini.

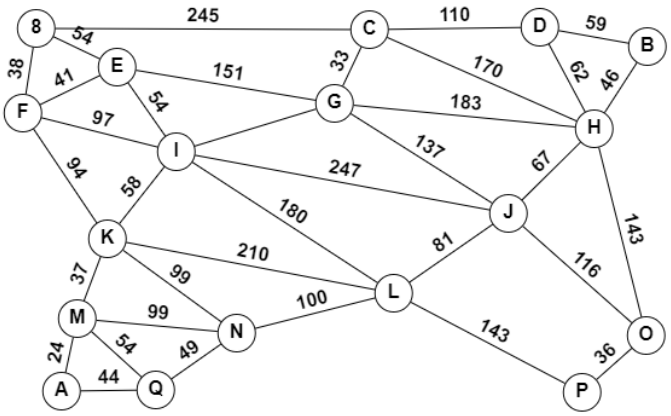
Gambar 16 merupakan bentuk graf berarah dalam Sirkuit *Hamilton*. Dapat dilihat bahwa setiap simpul graf dikunjungi tepat sekali kecuali titik 8 dan tidak ada sisi graf yang dilalui lebih dari sekali. Dengan setiap sisi berbobot satu, graf ini memiliki bobot total yaitu 19. Hal ini berarti bahwa Mobita membutuhkan 19 satuan waktu untuk mengantarkan seluruh barang yang ada dan kembali ke titik asal.

Dengan adanya sirkuit *Hamilton*. pemborosan langkah yang diambil dapat diminimalisasi sehingga pemain tidak perlu bolak-balik pada lokasi yang sama.

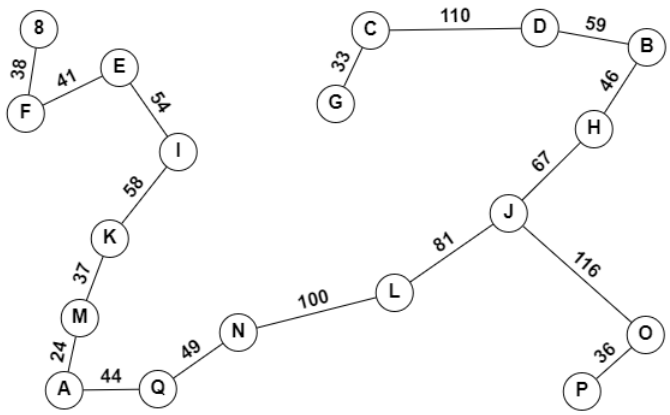


Gambar 16. Sirkuit Hamilton dari Gambar 15

Selain itu, pencarian rute dapat dilakukan dengan memanfaatkan pohon merentang minimum menggunakan algoritma *Prim*. Langkah ini diambil untuk mencoba pendekatan pohon merentang minimum untuk pencarian rute yang optimal dengan asumsi bobot sisi berpengaruh terhadap waktu perjalanan. Misalkan setiap sisi pada graf bobotnya direpresentasikan ke dalam satuan *pixel* dari gambar. Maka peta rute perjalanan Mobita akan menjadi seperti gambar 17. Langkah-langkah mencari pohon merentang minimum ditampilkan dalam gambar 19.



Gambar 17. Representasi Peta Perjalanan Mobita dengan Bobot Diambil dari Besar Pixel

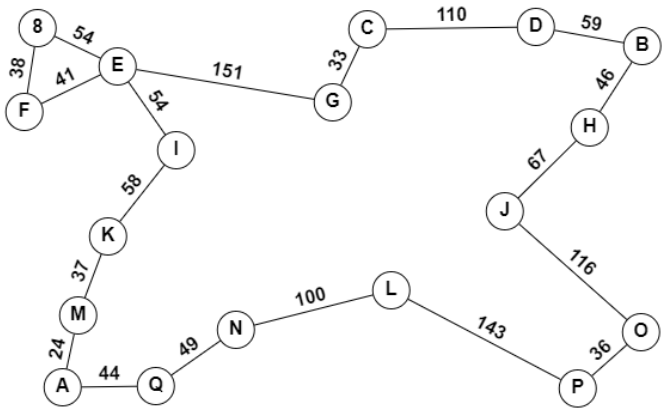


Gambar 18. Pohon Merentang Minimum dari Peta Mobita

Langkah	Sisi	Bobot	Pohon Rentang	Langkah	Sisi	Bobot	Pohon Rentang
1	(A, M)	24		10	(L, J)	143	
2	(M, K)	37		11	(J, O)	116	
3	(K, I)	58		12	(O, P)	36	
4	(I, E)	54		13	(J, H)	67	
5	(E, F)	41		14	(H, B)	46	
6	(F, 8)	38		15	(B, D)	59	
7	(A, Q)	44		16	(D, C)	110	
8	(Q, N)	49		17	(C, G)	33	
9	(N, L)	100					

Gambar 19. Langkah-Langkah Algoritma Prim

Pohon merentang minimum pada gambar 18 memiliki bobot total 993. Setelah didapat pohon merentang minimum, kemudian dimanfaatkan sirkuit *Hamilton* yang sudah ada untuk melengkapi graf pada gambar 18 agar menjadi sirkuit. Gambar 20 menampilkan hasil tersebut dengan bobot akhir graf menjadi 1260. Hasil ini didapat dengan menghapus sisi {L, J} dan menggantinya dengan sisi {L, P}. Kemudian menambahkan sisi {G, E} dan {E, 8}.



Gambar 20. Peta Rute Perjalanan Optimal

IV. KESIMPULAN

Dari analisis di atas dengan menggunakan teori sirkuit *Hamilton* dan algoritma *Prim* maka dapat dihasilkan rute perjalanan optimal yang dapat digunakan oleh Mobita untuk mengantarkan pesanan ke setiap titik antar hanya dengan satu siklus. Dengan adanya rute optimal ini, maka waktu yang diperlukan untuk menyelesaikan pengantaran barang dapat dilakukan seminimal mungkin. Dengan konfigurasi peta yang digunakan didapatkan waktu tempuh sebanyak 19 detik untuk menyusuri semua lokasi pengantaran. Konsep pohon merentang minimum yang digunakan untuk mencari rute perjalanan merupakan aproksimasi terhadap rute yang paling optimal.

Yang menjadi kekurangan dari algoritma ini adalah jika digunakan untuk mencari rute perjalanan dengan jumlah simpul dan sisi yang lebih banyak karena bobot yang dihasilkan akan semakin besar dan aproksimasi semakin jauh dari solusi terbaik yang bisa didapat. Untuk saat ini solusi yang ditampilkan adalah solusi dengan kompleksitas waktu yang masih normal walaupun hasilnya masih berupa aproksimasi terhadap hasil terbaik.

V. REFERENSI

[1] Munir, Rinaldi. 2021. Graf (Bagian 1). https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf. Diakses pada 11 Desember 2021.

[2] Munir, Rinaldi. 2021. Graf (Bagian 2). https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf. Diakses pada 11 Desember 2021.

[3] Munir, Rinaldi. 2021. Graf (Bagian 3). https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf. Diakses pada 11 Desember 2021.

[4] Munir, Rinaldi. 2021. Pohon (Bagian 1). https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag1.pdf. Diakses pada 11 Desember 2021.

[5] Munir, Rinaldi. 2021. Pohon (Bagian 2). https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag2.pdf. Diakses pada 11 Desember 2021.

[6] Baidoo, Evans. 2016. A Preliminary Study on Minimum Spanning Tree Algorithm Approach for Travelling Salesman Problem. https://www.researchgate.net/publication/308632016_A_Preliminary_Study_on_Minimum_Spanning_Tree_Algorithm_Approach_for_Travelling_Salesman_Problem. Diakses pada 12 Desember 2021.

[7] Ma, Suzanne. 2020. Understanding The Travelling Salesman Problem (TSP). https://blog.routific.com/travelling-salesman-problem. Diakses pada 12 Desember 2021.
[8] GeekforGeeks. 2021. Travelling Salesman Problem | Set 2 (Approximate using MST). https://www.geeksforgeeks.org/travelling-salesman-problem-set-2-approximate-using-mst. Diakses pada 13 Desember 2021.

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, 13 Desember 2021

Muhammad Fikri Ranjabi - 13520002