Aplikasi Pohon Merentang Minimum Untuk Optimisasi Pergerakan Armada dalam Permainan *Stellaris*

Fawwaz Anugrah Wiradhika Dharmasatya - 13520086[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]*author@itb.ac.id*

***Abstraksi*—*Stellaris* adalah permainan *grand strategy* bergenre *scifi* yang dikembangkan oleh Paradox Development Studio. Pada permainan ini, pemain memainkan sebuah peradaban yang baru saja mulai untuk menjelajahi bintang. Pada permainan ini, pemain bisa menjelajahi galaksi dan berinteraksi dengan peradaban lain. Peta permainan ini berupa sistem bintang yang dihubungkan dengan *hyperlane*. Karena ruang gerak pemain dibatasi oleh rute *hyperlane*, diperlukan cara agar pergerakan armada pemain menjadi efisien. Salah satu caranya dengan menggunakan pohon merentang minimum.**

***Keywords*— Graf, Pergerakan, *Stellaris*, Pohon.**

I. PENDAHULUAN

Gambar 1.1 Banner Permainan *Stellaris*
Sumber: https://stellaris.paradoxwikis.com/Stellaris diakses pada 12 Desember 2021

Stellaris adalah sebuah permainan video game yang dikembangkan oleh Paradox Development Studio dan dirilis pada tahun 2016. Pada permainan bergenre *grand strategy scifi* ini, pemain memainkan sebuah peradapan yang baru saja mulai untuk menjelajahi bintang-bintang. Di video game ini, pemain bisa melakukan banyak hal seperti menjelajahi bintang-bintang yang ada di galaksi, melakukan survey untuk menemukan sumber daya, dan berinteraksi dengan peradaban lain.

Di permainan ini, kunci dari penjelajahan antariksa adalah teknologi *hyperlane*. *Hyperlane* di permainan ini dideskripsikan sebagai sebuah jaringan yang menghubungkan bintang-bintang di galaksi. Jika sebuah pesawat bisa mengakses *hyperlane*, pesawat tersebut bisa melakukan perjalanan antar bintang dalam waktu yang sangat cepat melebihi kecepatan cahaya. Namun, *hyperlane* ini tidak tersebar merata dan ada kasus terdapat dua sistem bintang yang cukup berdekatan namun tidak terhubung dengan *hyperlane*.

Karena sifat *hyperlane* tersebut, pergerakan armada pemain menjelajahi bintang selama sebagian besar permainan harus mengikuti rute yang sudah terdefinisi di peta. Hal ini membuat perlunya sebuah strategi agar pergerakan armada tersebut efisien dan hanya menempuh rute *hyperlane* dengan jarak dan waktu sesedikit mungkin.

Peta permainan bisa dimodelkan sebagai sebuah graf. Simpul dari graf tersebut berupa sistem bintang dan sisi dari graf merupakan rute *hyperlane*. Karena itu, permasalahan rute yang efisien pada permainan ini bisa diselesaikan dengan teori graf serta konsep pohon yang merupakan turunan dari graf yang terdapat dalam Matematika Diskrit. Salah satu permasalahan yang bisa diselesaikan dengan teori graf adalah, permasalahan rute terefisien untuk menduduki semua sistem bintang musuh saat berperang yang bisa diselesaikan dengan menggunakan pohon merentang minimum.

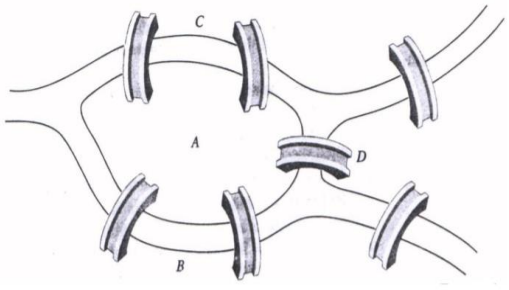
II. TEORI DASAR

A. Graf

1. Definisi Graf

   Graf adalah sebuah struktur yang terdiri dari pasangan dua buah himpunan, yakni himpunan simpul (*vertex*) yang tidak kosong serta himpunan sisi (*edge*). Simpul biasa direpresentasikan sebagai titik, sedangkan sisi direpresentasikan dengan sebuah garis yang menghubungkan dua buah simpul.

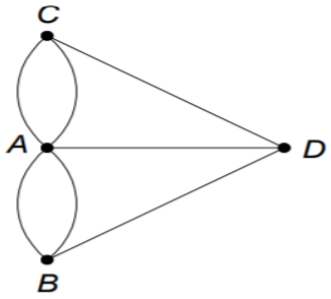
   Untuk lebih mengilustrasikan graf, berikut adalah Persoalan Jembatan Königsberg yang bisa direpresentasikan dengan graf:



Gambar 2.1 Persoalan Jembatan Königsberg

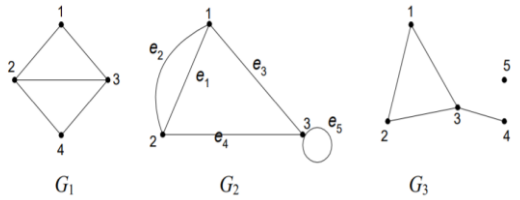
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf diakses pada 11 Desember 2021

Pada persoalan ini, tanah diantara sungai direpresentasikan dengan sebuah simpul dan setiap jembatan direpresentasikan sebagai sisi. Bentuk akhir graf dari permasalahan ini adalah sebagai berikut.



Gambar 2.2 Graf dari Persoalan Jembatan Königsberg
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf diakses pada 11 Desember 2021

2. Terminologi Dalam Graf



Gambar 2.3 Beberapa Contoh Graf Untuk Ilustrasi
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf diakses pada 11 Desember 2021

a) Ketetanggaan (*Adjacent*)

Dua buah simpul disebut bertetangga apabila mereka terhubung langsung oleh sebuah sisi. Pada graf G_1 dalam gambar 2.3 misalnya, simpul 1 terhubung langsung dengan simpul 2 dan 3, namun tidak terhubung langsung dengan simpul 4. Dengan kata lain, simpul 1 bertetangga dengan simpul 2 dan 3, namun tidak bertetangga dengan simpul 4.

b) Bersisian (*Incidency*)

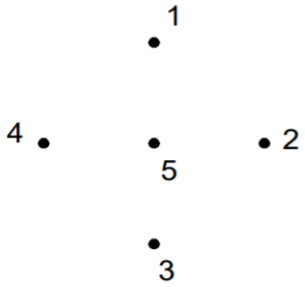
Bersisian bermakna bahwa sebuah sisi menghubungkan dua buah simpul. Contohnya adalah sisi e_1 pada graf G_2 di gambar 2.3 menghubungkan simpul 1 dan 2. Oleh karena itu, sisi e_1 disebut bersisian dengan simpul 1 dan simpul 2.

c) Simpul Terpencil (*Isolated Vertex*)

Sebuah simpul dikatakan terpencil apabila tidak ada satupun sisi yang bersisian dengannya. Contohnya adalah simpul 5 pada graf G_3 di gambar 2.3.

d) Graf Kosong (*Null Graph* atau *Empty Graph*)

Graf Kosong adalah graf yang himpunan sisinya merupakan himpunan kosong (N_N).



Gambar 2.4 Contoh Graf Kosong
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf diakses pada 11 Desember 2021

e) Derajat (*Degree*)

Derajat suatu simpul adalah jumlah sisi yang bersisian dengan simpul tersebut. Derajat direpresentasikan dengan *d(i)*, dengan i merupakan nama/nomor simpul. Dalam graf G_2 di gambar 2.3 misalnya, simpul 2 memiliki 3 buah sisi yang bersisian dengannya, yakni simpul e_1, e_2, dan e_4, sehingga derajat simpul 2 adalah 3 atau bisa ditulis d(2) = 3.

f) Lintasan(*Path*)

Lintasan yang panjangnya n dari simpul awal v_0 ke simpul tujuan v_n di dalam graf G ialah barisan berselang-seling simpul-simpul dan sisi-sisi yang berbentuk v_0, e_1, v_1, e_2, v_2, … , v_{n-1}, e_n, v_n sedemikian sehingga $e_1 = (v_0, v_1)$, $e_2 = (v_1, v_2)$, … , $e_n = (v_{n-1}. v_n)$ adalah sisi-sisi dari graf G. Panjang sebuah lintasan adalah jumlah sisi di lintasan tersebut. Contohnya pada graf G_1 di gambar 2.3. Lintasan 1, 2, 4, 3 adalah lintasan dengan barisan sisi (1, 2), (2, 4), (4,3) yang memiliki panjang lintasan sebesar tiga.

g) Siklus(*Cycle*) atau Sirkuit(*Circuit*)

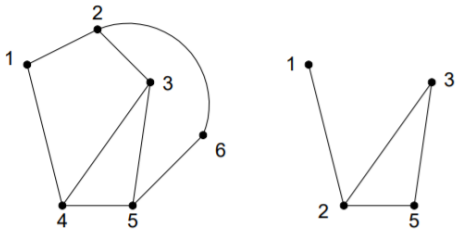
Siklus atau Sirkuit adalah lintasan yang berawal dan berakhir pada simpul yang sama. Contohnya pada graf G_1 di gambar 2.3. Lintasan 1, 2, 4, 3, 1 adalah sebuah sirkuit karena berawal dan berakhir di simpul 1. Lintasan ini memiliki panjang empat karena memiliki empat buah sisi, yakni (1,2), (2,4), (4,3), dan (3,1).

h) Keterhubungan(*Connected*)

Dua buah simpul v_1 dan v_2 dikatakan terhubung apabila terdapat lintasan yang berawal di v_1 dan berakhir di v_2. Graf G disebut graf terhubung(*connected graph*) apabila untuk setiap pasang simpul v_i dan v_j dalam himpunan V terdapat lintasan dari v_i ke v_j . Jika minimal ada sepasang simpul v_i dan v_j yang tidak memiliki lintasan dari v_i ke v_j , maka graf tersebut merupakan graf tak-terhubung(*disconnected graph*). Contoh graf terhubung adalah graf G_1 dan G_2 pada gambar 2.3, sedangkan contoh graf tak-terhubung adalah graf G_3 pada gambar 2.3.

i) Upagraf(*Subgraph*)

Misalkan G= (V, E) adalah sebuah graf.G_1= (V_1 , E_1) adalah upagraf dari G jika $V_1 \subseteq V$ dan $E_1 \subseteq E$.

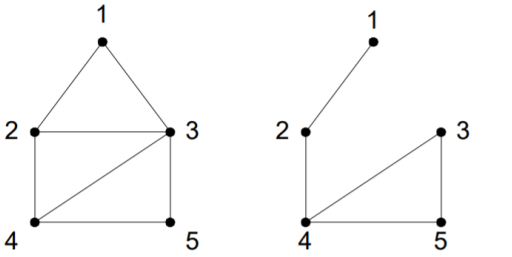


(a) Graf G_1 (b) Sebuah upagraf

Gambar 2.5 Contoh Sebuah Upagraf
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf diakses pada 11 Desember 2021

j) Upagraf Merentang(*Spanning Subgraph*)

Upagraf G1 = (V_1, E_1) dikatakan upagraf merentang dari G = (V, E) apabila G_1 mengandung semua simpul dari G.

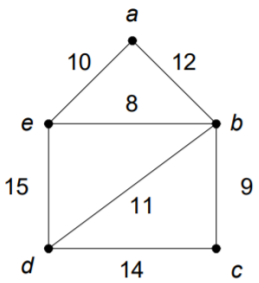


(a) graf G, (b) upagraf merentang dari G

Gambar 2.6 Contoh Upagraf Merentang
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf diakses pada 11 Desember 2021

k) Graf Berbobot(*Weighted Graph*)

Graf berbobot adalah graf yang setiap sisinya memiliki sebuah nilai/ bobot.

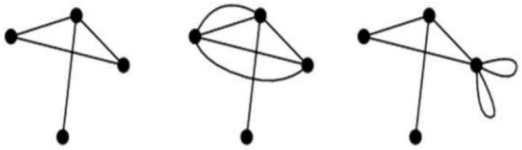


Gambar 2.7 Contoh Graf Berbobot
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf diakses pada 11 Desember 2021

3. Jenis Graf Berdasarkan Orientasi Arah Pada Sisi

1) Graf Tak-Berarah(*undirected graph*).

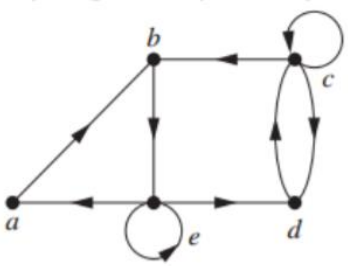
Graf tak-berarah adalah graf yang sisinya tidak mempunyai orientasi arah. Biasanya ditandai dengan sisi-sisinya berupa garis biasa.



Gambar 2.8 Contoh Graf Tak-Berarah
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf diakses pada 11 Desember 2021

2) Graf Berarah(*directed graph* atau *digraph*)

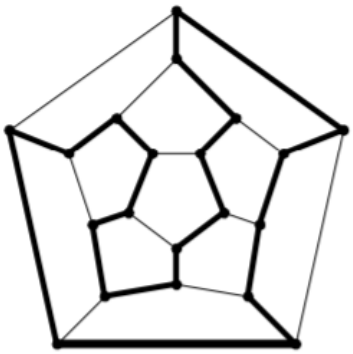
Graf berarah adalah graf yang setiap sisinya memiliki orientasi arah. Orientasi arah pada tiap sisi ini dilambangkan dengan ujung panah yang menunjukkan arah dari sisi terebut.



Gambar 2.9 Contoh Graf Berarah
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf diakses pada 11 Desember 2021

4. Graf Hamilton

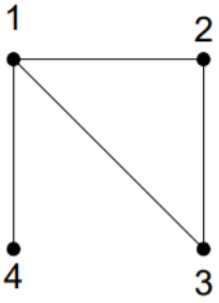
Graf Hamilton adalah graf yang memiliki Sirkuit Hamilton. Sirkuit Hamilton adalah sirkuit yang melewati tiap simpul pada graf tepat sekali, kecuali pada simpul asal yang dilewati dua kali.



Gambar 2.10 Contoh Graf Hamilton
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian3.pdf diakses pada 11 Desember 2021

5. Graf Semi-Hamilton

Graf Semi-Hamilton adalah graf yang memiliki Lintasan Hamilton. Lintasan Hamilton adalah lintasan yang melewati tiap simpul pada graf tepat sekali. Contohnya adalah lintasan 4-1-2-3 pada gambar 2.9 berikut ini.

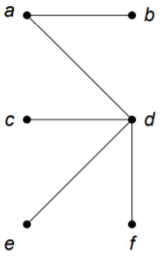


Gambar 2.11 Contoh Graf Semi-Hamilton
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian3.pdf diakses pada 11 Desember 2021

B. Pohon

1. Definisi Pohon

Pohon adalah graf tak-berarah terhubung yang tidak mengandung sirkuit.



Gambar 2.12 Contoh Pohon
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag1.pdf diakses pada 12 Desember 2021

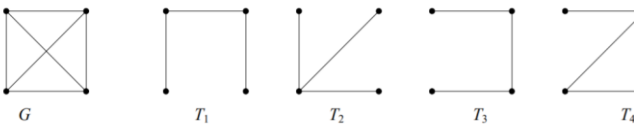
2. Sifat-Sifat Pohon

Misalkan G = (V, E) adalah graf tak-berarah sederhana dan jumlah simpulnya n. Maka:

1) G adalah pohon.
2) Setiap pasang simpul di dalam G terhubung dengan lintasan tunggal.
3) G terhubung dan memiliki n-1 buah sisi.
4) G tidak mengandung sirkuit.
5) Penambahan satu sisi pada graf G hanya akan menghasilkan satu sirkuit.
6) G terhubung dan semua sisinya adalah jembatan.
7)

3. Definisi Pohon Merentang

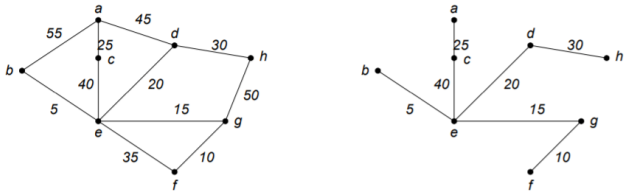
Pohon Merentang adalah upagraf merentang dari suatu graf terhubung yang berupa pohon. Pohon ini dapat diperoleh dengan memutus sirkuit di dalam graf. Setiap graf terhubung memiliki paling tidak sebuah pohon merentang.



Gambar 2.13 Sebuah Graf G dan Beberapa Pohon Merentangnya
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag1.pdf diakses pada 12 Desember 2021

4. Definisi Pohon Merentang Minimum

Sebuah graf terhubung yang berbobot mungkin memiliki lebih dari sebuah pohon merentang. Pohon Merentang Minimum adalah pohon merentang dari suatu graf berbobot terhubung yang memiliki bobot total terkecil. Bobot total ini diperoleh dengan menjumlahkan semua bobot/ nilai dari setiap sisi pada pohon merentang.



Gambar 2.14 Sebuah Graf Terhubung Berbobot Beserta Pohon Merentang Minimumnya
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag1.pdf diakses pada 12 Desember 2021

C. Algoritma Prim

Salah satu algoritma untuk mencari pohon merentang minimum dari suatu graf adalah Algoritma Prim. Langkah-langkah dalam algoritma ini adalah:

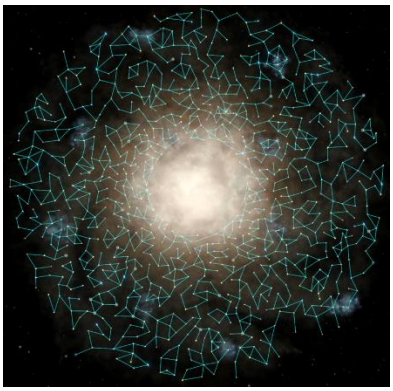
1) Ambil sisi dari graf G yang berbobot minimum, lalu masukkan ke sebuah himpunan T yang menyimpan sisi dalam bentuk pasangan simpul dari pohon merentang yang akan dibuat.
2) Pilih sisi (u, v) yang mempunyai bobot minimum dan bersisian dengan salah satu simpul di T, namun (u, v) tidak boleh sampai membentuk siklus di T. Lalu, masukkan (u, v) ke T.
3) Ulangi Langkah kedua sebanyak n-2 kali, yang mana n adalah jumlah simpul di graf G.

```
procedure Prim(input G : graf, output T : pohon)
{ Membentuk pohon merentang minimum T dari graf terhubung-
berbobot G.
Masukan: graf-berbobot terhubung G = (V, E), dengan |V|= n
Keluaran: pohon rentang minimum T = (V, E')
}
Deklarasi
  i, p, q, u, v : integer

Algoritma
  Cari sisi (p,q) dari E yang berbobot terkecil
  T ← {(p,q)}
  for i←1 to n-2 do
    Pilih sisi (u,v) dari E yang bobotnya terkecil namun
    bersisian dengan simpul di T
    T ← T ∪ {(u,v)}
  endfor
```

Gambar 2.15 *Pseudocode* Algoritma Prim
Sumber:
https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag1.pdf diakses pada 12 Desember 2021

D. Peta Permainan Stellaris



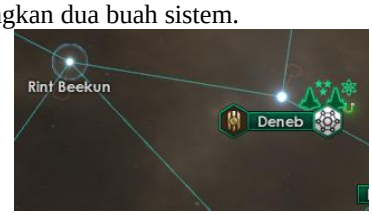
Gambar 2.16 Contoh Peta Permainan *Stellaris*
https://stellaris.paradoxwikis.com/Game_settings diakses pada 13 Desember 2021

Peta permainan *Stellaris* berupa sebuah galaksi yang terdiri dari banyak sistem bintang dan *hyperlane* yang menghubungkannya. Peta tersebut bisa dikonversi ke dalam bentuk graf tak-berarah berbobot dengan simpul dan sisi sebagai berikut.

1. Simpul
   Simpul dari peta ini adalah sistem-sistem bintang di dalamnya. Sistem ini bisa dimiliki oleh pemain, peradaban lain, atau tidak ada yang memilikinya sama sekali. Sistem hanya bisa dilewati pemain apabila peradaban yang memiliki tersebut tidak menutup perbatasan mereka dengan pemain.

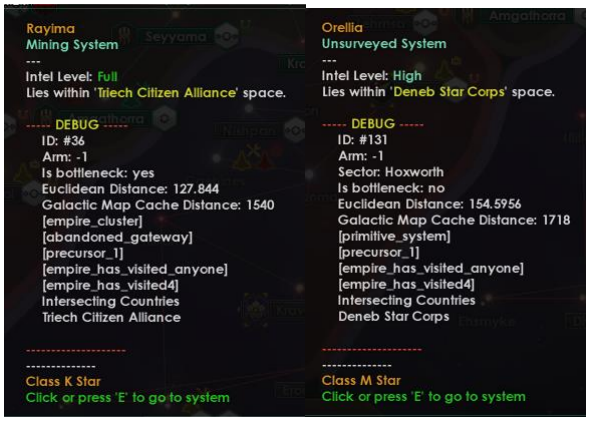
Gambar 2.17 Contoh Sistem di Peta
Sumber: Dokumentasi Penulis

2. Sisi
   Sisi dari peta ini adalah jalur *hyperlane* yang menghubungkan dua buah sistem.



Gambar 2.18 Contoh Sebuah *Hyperlane* yang Menghubungkan Sistem Deneb dengan Rint Beekun
Sumber: Dokumentasi Penulis

Jalur *hyperlane* ini memiliki panjang sehingga bila dikonversi ke graf akan mempunyai bobot. Data tentang panjang *hyperlane* dapat diperoleh dengan menghitung selisih panjang lintasan kedua simpul yang bersisian. Panjang lintasan ini dihitung dari mulai sistem pusat peradaban pemain hingga simpul yang dituju. Untuk mengakses data panjang lintasan ini, pertama perlu membuka *console command* lalu memasukkan perintah *debugtooltip*. Lalu gerakkan *cursor* ke sistem yang dihubungkan oleh *hyperlane* tersebut. Catat nilai di bagian Galactic Map Cache Distance(GMCD). Lalu ulangi untuk sistem lain yang bersisian dengan *hyperlane* tersebut. Setelah mendapatkan nilai kedua sistem yang bertetangga tersebut.



Gambar 2.19 Data *Debug* Untuk Mencari Panjang *Hyperlane* Antara Sistem Orellia dengan Rayima
Sumber: Dokumentasi Penulis

Sebagai Contoh, pada gambar 2.19 diperoleh nilai GMCD sistem Rayima adalah 1540 sedangkan nilai GMCD sistem Orellia sebesar 1718. Maka panjang *hyperlane* yang menghubungkan kedua sistem adalah:

$$|1718 - 1540| = 178$$

Nilai 178 ini yang akan menjadi bobot sisi antara simpul Rayima dan Orellia di graf.

III. PEMBAHASAN

A. Transformasi Peta ke Dalam Bentuk Graf

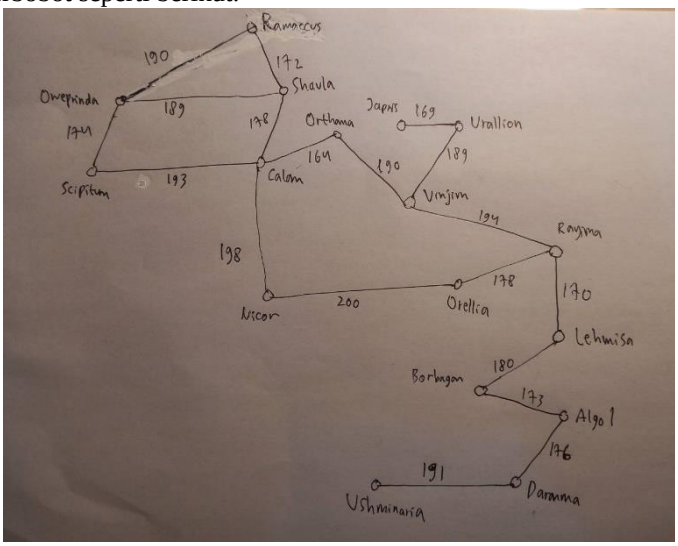
Gambar 3.1 Peta Permainan *Stellaris* yang Akan Dikonversi
Sumber: Dokumentasi Penulis

Langkah pertama untuk mengkonversi peta permainan ke dalam bentuk graf ialah menghilangkan simpul dan sisi di peta yang tidak menjadi target kita untuk dikunjungi. Misalnya pada gambar 3.1. Pada peta ini, pemain sedang memainkan negara dengan warna kecoklatan di bagian kanan atas peta. Pemain sedang dalam keadaan berperang dengan negara yang berwarna merah. Target pemain adalah menduduki setiap sistem bintang yang dimiliki negara berwarna merah. Oleh karena itu, perlu untuk menghilangkan sisi dan simpul yang terkoneksi atau dimiliki negara selain pemain dan musuhnya, atau simpul yang tidak dimiliki siapapun.

Karena pemain dan negara musuhnya hanya terhubung di sebuah simpul, yakni sistem bintang Rayima, maka wilayah perbatasan tersebut dimasukkan ke graf dan menjadi titik awal

untuk membentuk pohon merentang minimum. Wilayah pemain yang lain tidak perlu dimasukkan karena wilayah tersebut sudah dalam kontrol pemain.

Setelah itu, cari bobot tiap sisi pada graf yang terbentuk. Hasil konversi awal ini akan menghasilkan sebuah graf tak-berarah berbobot seperti berikut:

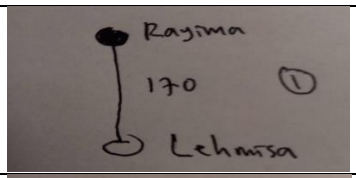
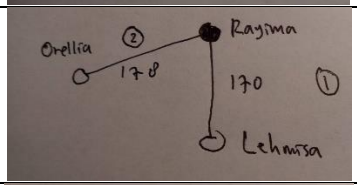
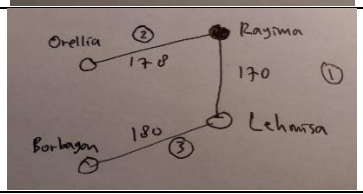
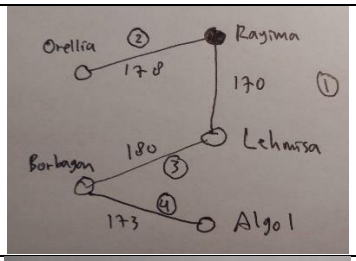
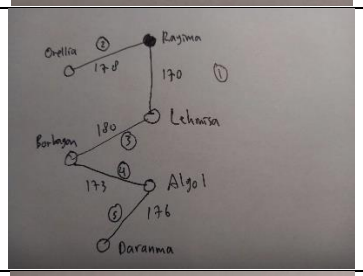
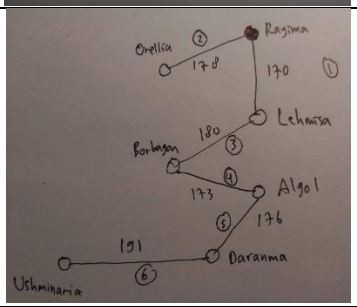
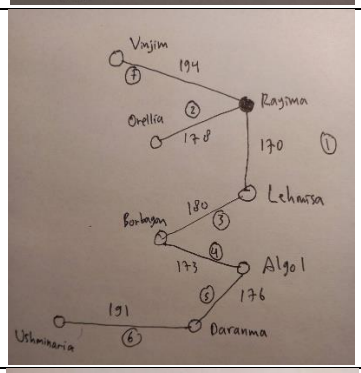
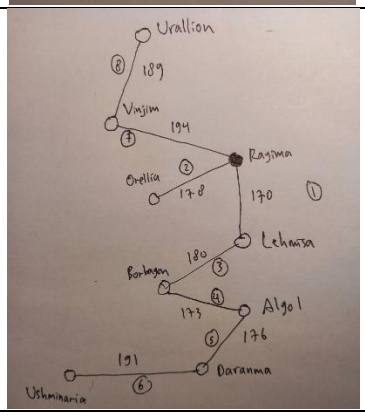


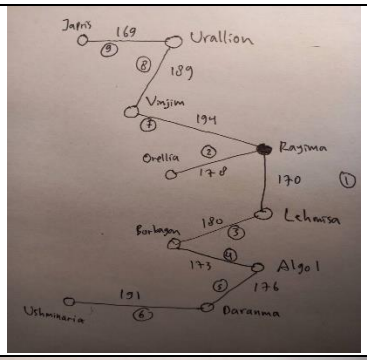
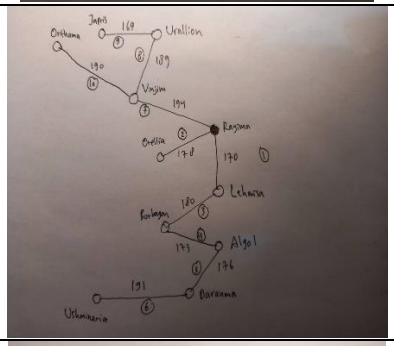
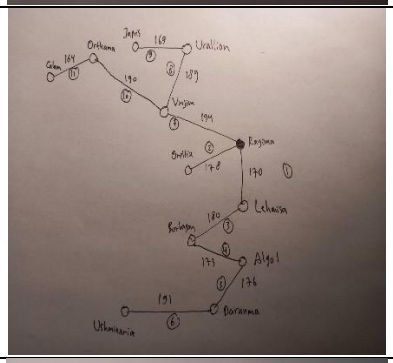
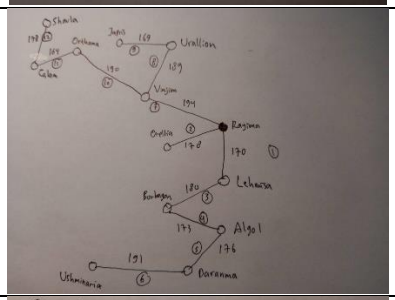
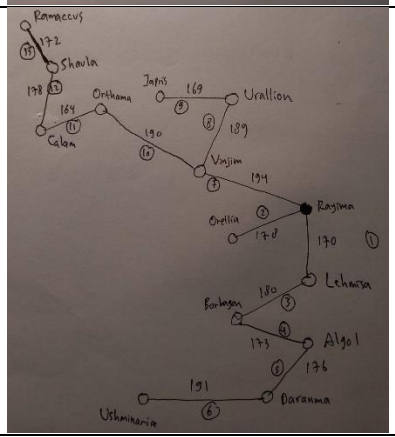
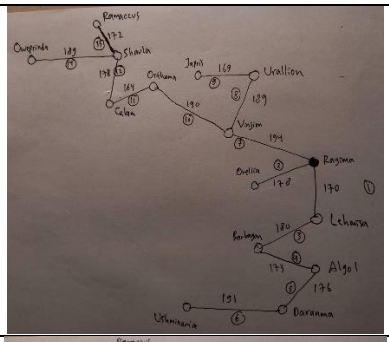
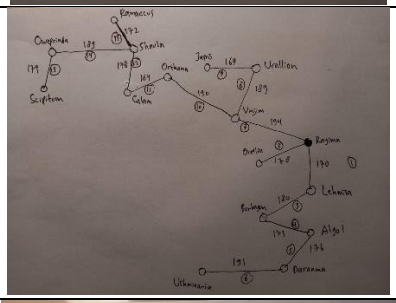
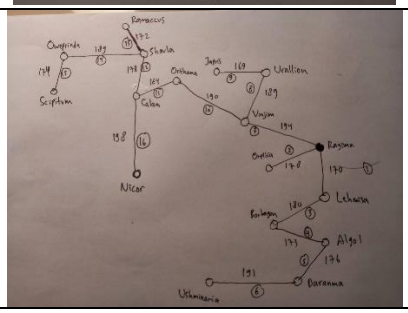
Gambar 3.2 Graf Hasil Konversi Peta
Sumber: Dokumentasi Penulis

B. Mencari Pohon Merentang Minimum

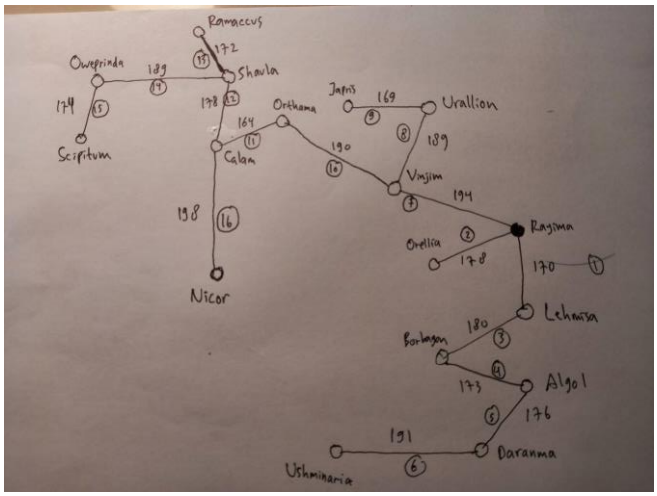
Pencarian pohon merentang minimum untuk mencari rute terpendek untuk mengunjungi semua simpul bisa ditemukan menggunakan algoritma Prim. Karena di contoh armada pemain berada di sistem Rayima, maka pembentukan pohon merentang dimulai dari simpul terebut.

Ilustrasi pembentukan pohon merentang minimum adalah sebagai berikut:

No.	Ilustrasi
1.	
2.	
3.	
4.	
5.	
6.	
7.	
8.	

9.	
10.	
11.	
12.	
13.	
14.	
15.	
16.	

Tabel 3.1 Langkah Per Langkah Pembuatan Pohon Merentang Minimum
Sumber: Dokumentasi Penulis



Gambar 3.3 Pohon Merentang Minimum yang Terbentuk
Sumber: Dokumentasi Penulis

Pohon merentang minimum yang telah terbentuk dapat digunakan sebagai gambaran pergerakan armada. Karena terdapat beberapa daun di pohon ini, armada yang ada bisa dibagi lagi ke dalam beberapa armada agar waktu untuk menduduki semua wilayah musuh menjadi lebih efisien.

IV. KESIMPULAN

Konsep pohon yang merupakan turunan teori graf memiliki banyak sekali kegunaan. Setiap hal yang bisa dimodelkan sebagai graf memiliki pohon merentang. Peta permainan *Stellaris* bisa dimodelkan sebagai graf sehingga memiliki pohon merentang. Pohon merentang minimum dapat digunakan untuk membentuk rute terpendek agar bisa mengunjungi semua sistem yang ingin dituju di permainan ini.

V. UCAPAN TERIMA KASIH

Puji syukur kepada Tuhan Yang Maha Esa, karena atas rahmat dan petunjuk-Nya penulis bisa menulis makalah ini. Penulis berterima kasih kepada semua pihak yang secara langsung maupun tidak langsung membantu dalam keberjalanan penulisan makalah ini. Penulis juga berterima kasih yang sebesar-besarnya kepada Bu Harlili selaku dosen pengampu Mata Kuliah Matematika Diskrit kelas K02. Bimbingan dan pengajaran beliau selama satu semester ini sangat membantu penulis untuk dapat menulis dan menyelesaikan makalah ini.

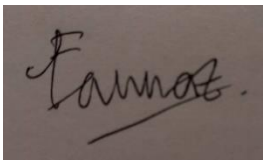
SUMBER

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf diakses pada 11 Desember 2021
[2] https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian3.pdf diakses pada 11 Desember 2021
[3] https://stellaris.paradoxwikis.com/Console_commands diakses pada 11 Desember 2021
[4] Y. Sulistyorini, *TEORI GRAPH*. Malang: Program Studi Pendidikan Matematika IKIP Budi Utomo Malang, 2018, pp. 1-2.
[5] https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag1.pdf diakses pada 12 Desember 2021
[6] https://stellaris.paradoxwikis.com/Stellaris diakses pada 12 Desember 2021
[7] https://stellaris.paradoxwikis.com/FTL diakses pada 12 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, 14 Desember 2021



Fawwaz Anugrah Wiradhika Dharmasatya 13520086