Aplikasi Algoritma Prim dalam Menentukan Rute Terpendek untuk Membuka Vault of Tartaros pada Permainan Immortals Fenyx Rising

Althaaf Khasyi Atisomya - 13521130[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]*13521130@mahasiswa.itb.ac.id*

***Abstract*—Immortals Fenyx Rising adalah *game* dengan tema *fantasy open-world adventure.* Salah satu aspek terpenting dalam permainan ini adalah eksplorasi dunia untuk mendapatkan perlengkapan dan mengalahkan musuh-musuh. Permainan ini sendiri memiliki tempat yang cukup luas sehingga untuk mengeksplorasinya membutuhkan waktu yang lama. Terdapat fitur dalam permainan ini yang memungkinkan pemain melakukan teleport salah satunya adalah dengan membuka Vault of Tartaros. Vault of Tartaros tidak hanya memungkinkan player untuk melakukan teleportasi tetapi juga dapat memberi player reward yang tentunya sangat berguna dalam game apabila diselesaikan. Dengan memanfaatkan pohon merentang minimum dengan algoritma prim kita dapat mengetahui lintasan rute terpendek dalam membuka Vault of Tartaros dalam permainan Immortals Fenyx Rising.**

***Keywords*—Pohon Merentang Minimum, Algoritma Prim, Immortals Fenyx Rising, Vault of Tartaros.**

I. PENDAHULUAN

Immortals Fenyx Rising merupakan permainan yang mengusung genre *action-adventure* bertemakan mitologi Yunani yang diluncurkan oleh Ubisoft, developer yang telah sering membuat permainan *open-world*. Permainan ini menceritakan tentang seorang manusia setengah dewa, bernama Fenyx, yang kini adalah manusia terakhir yang tersisa di Yunani. Pada permainan ini semua manusia telah berubah menjadi batu karena ulah dewa jahat bernama Typhon. Akibatnya, Fenyx harus mengalahkan Typhon dan berusaha untuk mengembalikan dunia seperti semula.

Gambar 1. Demo permainan Immortals Fenxy Rising
(Sumber: https://www.nintendo.com/store/products/immortals-fenyx-rising-switch/ diakses pada 9 Desember 2022 pukul 21.33)

Pada awal permainan, pemain belum memiliki kemampuan maupun perlengkapan apa pun sehingga kita diharuskan untuk melawan musuh, memecahkan *puzzle*, dan mengeksplorasi dunia yang ada di permainan ini untuk mencari perlengkapan dan *skill* demi mengalahkan Typhon. Dunia pada permainan ini sendiri terbagi menjadi tujuh wilayah berbeda yang masing-masing dapat memberikan perlengkapan dan kekuatan baru bagi Fenyx. Salah satu cara untuk mendapatkan perlengkapan dan *skill* adalah dengan melakukan *challenge* Tartaros.

Gambar 2. Gameplay permainan Immortals Fenyx Rising
(Sumber: https://www.vg247.com/immortals-fenyx-rising-aphrodite-tears diakses pada 9 Desember 2022 pukul 21.38)

Vault of Tartaros merupakan salah satu *challenge* yang harus dilakukan agar alur cerita dapat berjalan. Menyelesaikan Vault of Tartaros akan memberi pemain hadiah berupa senjata maupun *armor* yang tentunya akan memperkuat pemain dan memungkinkan pemain untuk mengupgrade stamina. Selain memberi perlengkapan dan skill membuka Vault of Tartaros juga memungkinkan pemain untuk berteleportasi di mana pun pemain berada. Untuk membuka kemampuan teleportasi Vault of Tartaros pemain tidak perlu menyelesaikan *challenge* dan *puzzle* yang ada, hanya dengan memasukinya saja pemain telah dapat melakukan transportasi ke Vault of Tartaros tersebut. Hal ini tentunya sangat membantu dalam proses eksplorasi mengingat dunia g*ame* yang sangat luas ini.

Untuk menghemat waktu dalam mengakses Vault of Tartaros diperlukan rute terpendek yang dapat membantu pemain untuk membuka semua Vault of Tartaros dalam waktu singkat. Pada makalah ini penulis akan mencari rute tersingkat dengan memanfaatkan aplikasi pohon merentang dengan algoritma prim.

II. DASAR TEORI

A. Graf

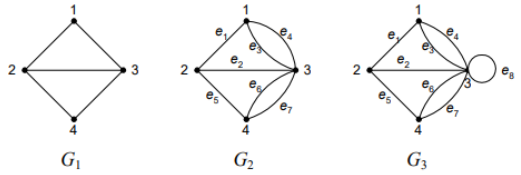
Graf adalah struktur objek yang terdiri dari simpul (*vertex*) dan sisi (*ege*) yang masing-masing simpul memiliki keterhubungan atau keterkaitan tertentu yang dihubungkan oleh sisinya. Secara formal graf dapat direpresentasikan oleh :

$$G = (V, E)$$

G = graf
V = himpunan tak kosong dari simpul-simpul
$=\{v_1,v_2,v_3,\ldots,v_n\}$
E = himpunan sisi yang menghubungkan dua simpul
$= \{e_1,e_2,e_3,\ldots,e_n\}$

Berdasarkan ada tidaknya sisi ganda dan gelang, graf dapat digolongkan menjadi dua jenis yaitu:

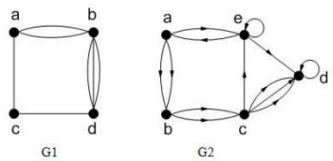
1. Graf sederhana
   Graf yang tidak memiliki sisi ganda maupun gelang.
2. Graf tak-sederhana
   Graf yang mengandung sisi ganda atau gelang. Graf ini dapat dibagi lagi menjadi graf ganda dan graf semu. Graf ganda adalah graf yang mengandung sisi ganda tetapi tidak mengandung gelang, sedangkan graf semu mengandung gelang.



Gambar 3. (a) graf sederhana (b) graf ganda (c) graf semu
(Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf diakses 9 Desember pukul 22.34)

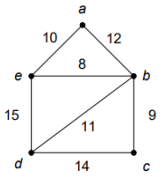
Berdasarkan ada tidaknya orientasi pada sisi, graf dibedakan menjadi:

1. Graf tak berarah
   Graf yang sisinya tidak memiliki orientasi arah.
2. Graf berarah
   Graf yang setiap sisinya memiliki orientasi arah



Gambar 4.(a) graf tak berarah (b) graf berarah
(Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf diakses 9 Desember pukul 22.34)

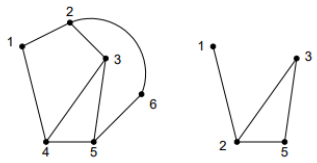
Suatu graf dapat memiliki bobot yaitu suatu harga yang ada pada setiap sisi. Graf yang memiliki bobot ini disebut graf berbobot. Harga pada graf ini dapat merepresentasikan suatu nilai, misalnya jarak dua simpul, waktu tempuh, dan biaya perjalanan. Pada makalah ini akan digunakan graf berbobot yang nilainya merepresentasikan jarak dua simpul.



Gambar 5.Graf berbobot
(Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf diakses 9 Desember pukul 22.41)

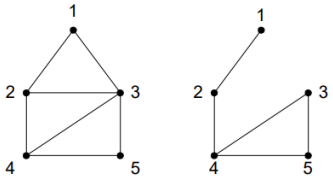
Graf juga memiliki beberapa terminologi, beberapa terminologi graf yang akan digunakan pada makalah ini, yaitu:

1. Ketetanggaan (*Adjacent*)
   Dua buah simpul pada graf dikatakan bertetangga apabila keduanya terhubung langsung oleh suatu sisi.
2. Bersisian (*Incidency*)
   Sebuah simpul dan sebuah sisi dikatakan bersisian apabila sisi tersebut menghubungkan simpul dengan sembarang simpul lain.
3. Lintasan (*Path*)
   Lintasan simpul awal dan simpul akhir dalam graf adalah barisan berselang-seling simpul dan sisi.
4. Siklus (*Cycle*) atau Sirkuit (*Circuit*)
   Siklus adalah lintasan yang berawal dan berakhir pada simpul yang sama.
5. Upagraf (*Subgraf*)
   Graf G1 = (V_1, E_1) adalah upagraf dari G = (V, E) jika dan hanya jika $V_1 \subseteq V$ dan $E_1 \subseteq E$.



Gambar 6. Sebuah graf dan salah satu Upagraf nya
(Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian2.pdf diakses pada 9 Desember 22.55)

6. Upagraf Merentang
   Graf G1 = (V_1, E_1) merupakan upagraf merentang dari graf G = (V, E) jika $V_1 = V$.

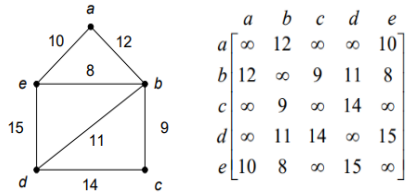


Gambar 7. Sebuah graf dan salah satu Upagraf Merentang nya
(Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian2.pdf diakses pada 9 Desember 22.55)

Suatu graf tidak hanya dapat direpresentasikan dengan diagram, pada makalah ini penulis akan menggunakan representasi graf sebagai matriks ketetanggaan dan juga graf sebagai senarai ketetanggaan.

1. Matriks ketetanggaan
   Misalkan G = (V, E) adalah suatu graf dengan banyaknya simpul V adalah $\{v_1, v_2, v_3,\ldots,v_n\}$. Maka

matriks ketetanggaan A dari G adalah matriks ukuran n x n di mana A_{ij} bernilai 1 apabila simpul v_i dan v_j bertetangga dan bernilai 0 apabila simpul v_i dan v_j tidak bertetangga. Pada graf berbobot nilai A_{ij} merepresentasikan bobot dari sisi yang bersisian dengan kedua simpul tersebut.

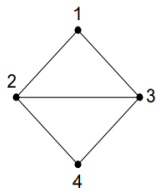


	a	b	c	d	e
a	∞	12	∞	∞	10
b	12	∞	9	11	8
c	∞	9	∞	14	∞
d	∞	11	14	∞	15
e	10	8	∞	15	∞

Gambar 8. Representasi matriks ketetanggaan pada graf berbobot
(Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian2.pdf diakses pada 9 Desember 22.55)

2. Senarai ketetanggaan

Matriks ketetanggaan kurang efektif dalam merepresentasikan graf yang memiliki sisi yang sedikit (graf tidak padat) sehingga untuk mengatasi masalah ini digunakan Senarai ketetanggaan. Senarai ketetanggaan berisi list dari simpul yang bertetangga dengan suatu simpul.

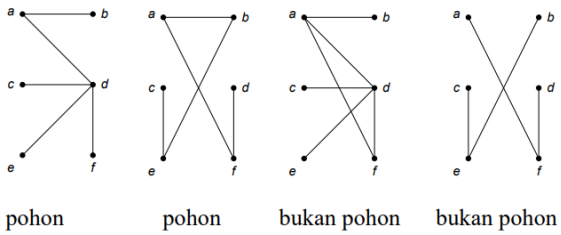


Simpul	Simpul Tetangga
1	2, 3
2	1, 3, 4
3	1, 2, 4
4	2, 3

Gambar 9. Representasi senarai ketetanggaan
(Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian2.pdf diakses pada 9 Desember 22.55)

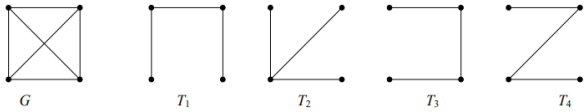
B. Pohon

Pohon merupakan suatu graf tak berarah sederhana yang tidak mengandung sirkuit. Ini berarti, pohon merupakan graf yang memiliki sifat-sifat khusus, yaitu setiap pasang simpul dalam pohon terhubung dengan lintasan yang tunggal, pohon memiliki m = n-1 buah sisi, dan tidak boleh mengandung sirkuit.



Gambar 10. Pohon
(Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag1.pdf diakses pada tanggal 9 Desember 23.41)

Pohon merentang dari graf merupakan upagraf merentang yang berupa pohon. Pohon merentang ini diperoleh dengan memutus sirkuit di dalam graf sehingga graf yang memiliki k komponen akan mempunyai k buah pohon merentang. Pohon merentang memiliki berbagai macam aplikasi dalam kehidupan, contohnya jumlah ruas jalan minimum yang mungkin untuk menghubung kan semua tempat sehingga tempat tetap saling terhubung.



Gambar 11. Pohon dan k pohon merentangnya
(Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag1.pdf diakses pada tanggal 9 Desember 23.23)

Pohon merentang minimum adalah pohon merentang yang memiliki bobot minimum pada graf berbobot. Pada makalah ini penulis akan memanfaatkan pohon merentang minimum untuk menentukan lintasan terpendek yang dapat dilalui untuk membuka semua Vault of Tartaros. Menentukan pohon merentang minimum pada suatu graf berbobot dapat dilakukan dengan berbagai cara pada makalah ini penulis akan menggunakan algoritma prim.

C. Algoritma Prim

Algoritma prim adalah sebuah algoritma untuk mencari pohon merentang minimum suatu graf berbobot. Algoritma prim memiliki Langkah-langkah sebagai berikut:

1. Ambil sisi (u0, v0) graf G yang memiliki bobot minimum dan masukkan ke dalam graf hasil.
2. Masukkan sisi (u,v) ke graf hasil yang mempunyai bobot minimum dan bersisian dengan simpul yang telah ada pada graf hasil dan tidak akan membentuk sirkuit.
3. Ulangi Langkah 2 sebanyak jumlah simpul-2

```
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 12. Pseudocode Algoritma Prim
(Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag1.pdf diakses pada tanggal 9 Desember 23.41)

III. HASIL DAN PEMBAHASAN

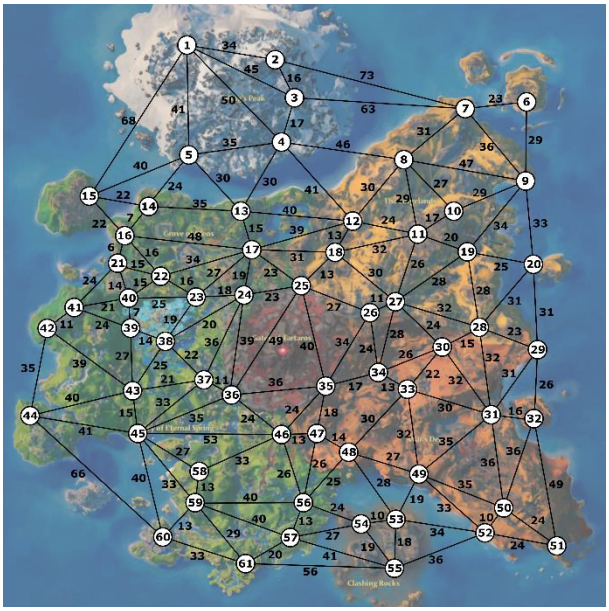
A. Pembuatan Graf

Gambar 13. Map Immortals Fenyx Rising
(Sumber: https://mapgenie.io/immortals-fenyx-rising/maps/golden-isle diakses pada 10 Desember 00.10)

Dalam menentukan lintasan minimum yang harus ditempuh untuk membuka Vault of Tartaros terlebih dahulu kita membuat graf sederhana tak berarah yang memiliki bobot berdasarkan jarak dua tempat. Dalam permainan Immortals Fenyx Rising sendiri terdapat 61 buah vault entrance yang masing-masing akan kita buat sebagai simpul dalam graph (gambar 14).

Gambar 14. Map Immortals Fenyx Rising dengan Vault of Tartaros direpresentasikan sebagai simpul dalam graf
(Sumber: dokumen pribadi)

Setelah mendapatkan simpul graf kita perlu menghubungkan simpul-simpul tadi dan menghitung jarak antara dua simpul. Sayangnya game Immortals Fenyx Rising tidak menyediakan informasi mengenai jarak antara dua vault entrances. Oleh karena itu, dalam menentukan jarak dua simpul penulis memanfaatkan aplikasi Inkscape yang mengukur jarak dua titik berdasarkan jarak dua titik pada gambar.



Gambar 15. Diagram graf jarak antara dua vault entrances
(Sumber: dokumen pribadi)

Selain direpresentasikan dalam bentuk diagram (gambar 15), graf juga direpresentasikan dalam bentuk senarai ketetanggaan untuk memudahkan proses pengkodingan (gambar 16). Format senarai ketetanggaan yang digunakan adalah sebuah array dengan panjang 61 (jumlah simpul) yang tiap elemennya merupakan array dari simpul yang bertetangga dengan index dari array serta bobotnya contoh [2,34,3,45,4,50,5,41,15,68] pada index 0 memiliki arti bahwa simpul v_1 bertetangga dengan v_2, v_3, v_4, v_5, dan v_{15} dengan bobot masing-masing secara berturut adalah 34, 45, 50, 41, dan 68.

Gambar 16. Senarai ketetanggaan graf jarak antara dua vault entrances
(Sumber: dokumen pribadi)

Senarai ketetanggaan memang lebih efektif dalam merepresentasikan graf jarak antara dua vault entrances tetapi dalam pembuatan algoritma prim penggunaan matriks ketetanggaan lebih mudah digunakan. Oleh karena itu perlu kode yang mengubah senarai ketetanggaan menjadi matriks ketetanggaan (gambar 17).

```python
def createGraph(fileName,V):
    # mengembalikan graph yang direpresentasikan
    # sebagai matriks ketetanggaan

    # membaca senarai ketetanggaan dari file
    raw = open(fileName,'r')
    raw = raw.read().splitlines()[1:]
    data = []
    for i in range(len(raw)):
        data.append(raw[i].split(','))
        for j in range(len(data[i])):
            data[i][j] = int(data[i][j])

    # mengubah senarai ketetanggaan menjadi
    # matrix ketetanggaan
    graph = [[0 for j in range(V)] for i in range(V)]
    for i in range(len(data)):
        for j in range(0,len(data[i]),2):
            graph[i][data[i][j]-1] = data[i][j+1]

    return graph
```

Gambar 17. Kode mengubah senarai ketetanggaan menjadi matriks ketetanggaan dalam python
(Sumber: dokumen pribadi)

B. Pembuatan Algoritma Prim

```python
def primAlgorithm(G):
    # mencetak langkah-langkah algoritma prim
    # sampai ditemukan pohon merentang min dari G

    V = len(G)
    selected = [False for i in range(V)]

    # mencari sisi (v_min,v_lain) pada G dengan bobot terkecil
    min = 99999
    v_min = -1
    for i in range(V):
        for j in range(V):
            if min>G[i][j] and G[i][j]:
                min = G[i][j]
                v_min = i
    selected[v_min] = True

    counter = 0
    while (counter <= V-2):
        # pilih sisi G yang memiliki bobot terkecil dan
        # bersisian dengan simpul yang telah di selected
        v_awal = 0
        v_akhir = 0
        min = 99999
        for i in range(V):
            if selected[i]:
                for j in range(V):
                    if ((not selected[j]) and G[i][j]):
                        if min > G[i][j]:
                            min = G[i][j]
                            v_awal = i
                            v_akhir = j
        selected[v_akhir] = True
        display(counter,v_awal,v_akhir)
        counter += 1

if __name__ == "__main__":
    G = createGraph('raw.csv',61)
    primAlgorithm(G)
```

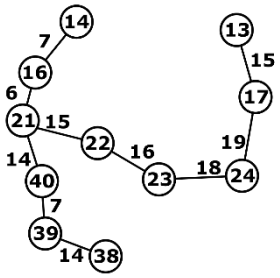
Gambar 18. Kode algoritma prim dalam python
(Sumber: dokumen pribadi)

Algoritma prim (gambar 18) diperlukan dalam menentukan pohon merentang minimum. Langkah pertama yaitu mencari simpul yang memiliki bobot terkecil dan memasukkannya pada array selected (array of boolean sepanjang banyak simpul yang bernilai true jika simpul telah masuk dalam graf merentang yang akan dihasilkan, false jika belum). Kemudian, dilakukan iterasi sebanyak jumlah simpul-2 untuk memasukkan sisi yang memiliki bobot terkecil dan bersisian dengan simpul yang telah di selected sehingga dihasilkan pohon merentang minimum.

C. Hasil

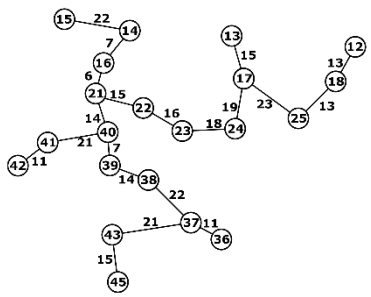
Program akan mencetak output yang merupakan Langkah-langkah dari algoritma prim untuk menghasilkan pohon merentang minimum.

No.	Sisi	Bobot
1.	16-21	6
2.	16-14	7
3.	21-40	14
4.	40-39	7
5.	39-38	14
6.	21-22	15
7.	22-23	16
8.	23-24	18
9.	24-17	19
10.	17-13	15



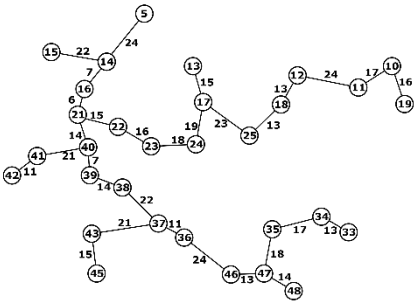
Gambar 19. Langkah 1-10 Algoritma prim
(Sumber: dokumen pribadi)

No.	Sisi	Bobot
11.	40-41	21
12.	41-42	11
13.	14-15	22
14.	38-37	22
15.	37-36	11
16.	37-43	21
17.	43-45	15
18.	17-25	23
19.	25-18	13
20.	18-12	13



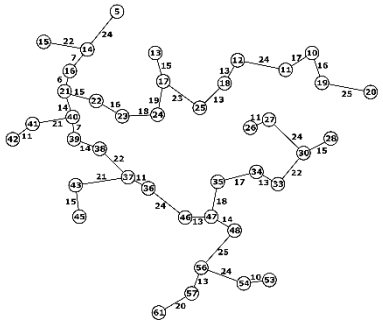
Gambar 20. Langkah 11-20 Algoritma prim
(Sumber: dokumen pribadi)

No.	Sisi	Bobot
21.	12-11	24
22.	11-10	17
23.	10-19	16
24.	14-5	24
25.	36-46	24
26.	46-47	13
27.	47-48	14
28.	47-35	18
29.	35-34	17
30.	34-33	13



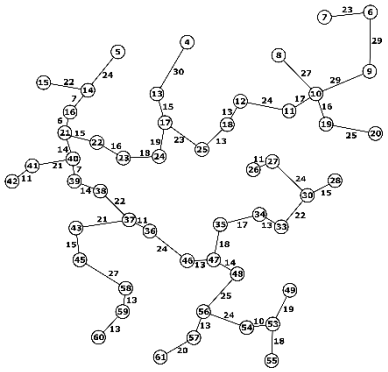
Gambar 21. Langkah 21-30 Algoritma prim
(Sumber: dokumen pribadi)

No.	Sisi	Bobot
31.	33-30	22
32.	30-28	15
33.	30-27	24
34.	27-26	11
35.	19-20	25
36.	48-56	25
37.	56-57	13
38.	57-61	20
39.	56-54	24
40.	54-53	10



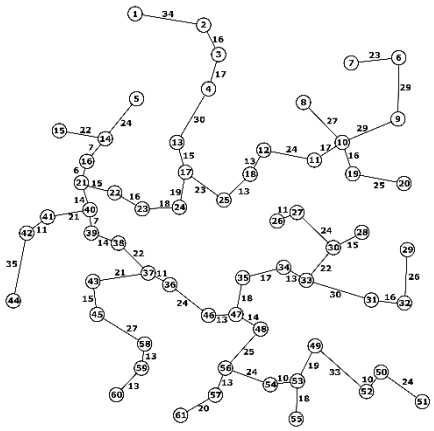
Gambar 22. Langkah 31-40 Algoritma prim
(Sumber: dokumen pribadi)

No.	Sisi	Bobot
41.	53-55	18
42.	53-49	19
43.	10-8	27
44.	45-58	27
45.	58-59	13
46.	59-60	13
47.	10-9	29
48.	9 -6	29
49.	6 -7	23
50.	13-4	30



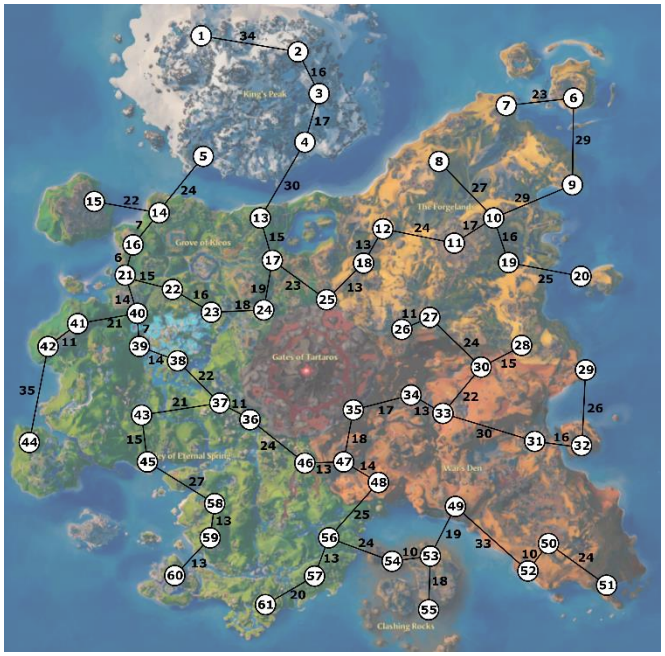
Gambar 23. Langkah 41-50 Algoritma prim
(Sumber: dokumen pribadi)

No.	Sisi	Bobot
51.	4 -3	17
52.	3 -2	16
53.	33-31	30
54.	31-32	16
55.	32-29	26
56.	49-52	33
57.	52-50	10
58.	50-51	24
59.	2 -1	34
60.	42-44	35



Gambar 24. Langkah 51-60 Algoritma prim
(Sumber: dokumen pribadi)

Pohon merentang minimum inilah (gambar 25) yang merupakan lintasan terpendek untuk membuka Vault of Tartaros dalam game Immortals Fenyx Rising.



Gambar 25. Lintasan minimum yang ditempuh untuk membuka semua Vault of Tartaros
(Sumber: dokumen pribadi)

IV. KESIMPULAN

Aplikasi teori graf dan pohon memiliki banyak manfaat yang dapat diaplikasikan dalam kehidupan. Penggunaan algoritma prim terbukti dapat menentukan lintasan terdekat dalam membuka Vault of Tartaros. Dengan merepresentasikan jarak antara vault entrances menjadi graf berbobot, jalan terdekat untuk membuka Vault of Tartaros dapat ditentukan. Lintasan terdekat dalam membuka Vault of Tartaros dapat dilihat pada gambar 21. Hasil dari makalah ini diharapkan dapat membantu pemain untuk berprogres dalam permainan Immortals Fenyx Rising dan memudahkan pemain untuk menyelesaikan permainan ini.

V. UCAPAN TERIMAKASIH

Puji syukur penulis panjatkan kepada Allah Swt. karena atas rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan makalah yang berjudul “Aplikasi Algoritma Prim untuk Menentukan Rute Terpendek dalam Membuka Vault of Tartaros pada Permainan Immortals Fenyx Rising”. Makalah ini dibuat untuk memenuhi tugas akhir semester I pada mata kuliah Matematika Distrik IF2120.

Makalah ini tidak akan terwujud tanpa adanya bantuan, arahan, dan dorongan dari berbagai pihak. Untuk itu, penulis menyampaikan terima kasih kepada orang tua penulis serta Ibu Dr. Fariska Zakhralativa Ruskanda, S.T., M.T. selaku dosen pengampu mata kuliah Matematika Distrik IF2120.

REFERENSI

[1] Demo Permainan Immortals Fenyx Rising. Diakses 9 Desember 2022 pukul 21.33 dari https://www.nintendo.com/store/products/immortals-fenyx-rising-switch/

[2] Gameplay Immortals Fenyx Rising. Diakses 9 Desember 2022 pukul 21.38 dari https://www.vg247.com/immortals-fenyx-rising-aphrodite-tears

[3] Map Immortals Fenyx Rising. diakses pada 10 Desember 2022 pukul 00.10 dari https://mapgenie.io/immortals-fenyx-rising/maps/golden-isle

[4] Munir, Rinaldi. 2022. *Graf (Bagian 1).* Diakses 10 Desember pukul 22.34 dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf

[5] Munir, Rinaldi. 2022. *Graf (Bagian 2).* Diakses 10 Desember 2022 pukul 22.55 dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian2.pdf

[6] Munir, Rinaldi. 2022. *Pohon (Bagian 1).* Diakses 10 Desember 2022 pukul 23.41 dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Pohon-2020-Bag1.pdf

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, 10 Desember 2022

Althaaf Khasyi Atisomya/13521130