Penggunaan Algoritma Dijkstra untuk Mengoptimalisasi Rute Pengumpulan *Starconch* dalam Permainan Genshin Impact

Axel Santadi Warih - 13522155
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13522155@mahasiswa.itb.ac.id

Algoritma Dijkstra, dikenal dalam teori graf sebagai algoritma pencarian jalur terpendek, diadaptasi untuk memodelkan dan menentukan jalur optimal dalam konteks pengumpulan Starconch. Penelitian ini menggabungkan aspek teoritis algoritma Dijkstra dengan implementasi praktisnya dalam konteks permainan Genshin Impact, dengan hasil eksperimen dan evaluasi performa menunjukkan peningkatan signifikan dalam efisiensi pengumpulan Starconch dibandingkan dengan pendekatan tanpa algoritma. Dengan demikian, makalah ini memberikan wawasan dan kontribusi penting untuk pemahaman dan pengembangan strategi pengoptimalan dalam permainan dunia terbuka yang kompleks seperti Genshin Impact.

Kata kunci: Genshin Impact, Starconch, algoritma Dijkstra, pengoptimalan rute, efisiensi permainan.

I. PENDAHULUAN

Genshin Impact merupakan sebuah game yang sangat populer dalam berbagai kalangan masyarakat dalam waktu yang belum lama ini. Permainan ini pertama kali dibuat oleh miHoYo yang dimana berpindah tangan kepada Cognosphere Pte. Ltd. Yang berada dibawah merek HoYoverse, dimana game ini rilis pada 28 September 2020 dengan format versi 1.0 bernama "Welcome To Teyvat." Di dalam permainan ini terdapat banyak *playable character* yang dapat dipilih oleh player untuk dikendalikan di dalam dunia yang sangat luas. Salah satu hal yang perlu player lakukan untuk memperkuat karakter mereka, adalah melakukan level up dan *ascending* karakter mereka. Untuk melakukan level up, player tidak memerlukan banyak bahan dalam permainan, cukup hanya buku *exp*, dan juga uang dalam game, tetapi untuk melakukan *ascending*, player diharuskan mengumpulkan lebih banyak bahan, salah satu bahan yang diperlukan bernama *Local Specialities*.

Salah satu bahan dari *Local Specialities* ini adalah *Starconch* Dimana karakter yang membutuhkan bahan tersebut Bernama Childe dan juga Yelan. Bahan ini hanya bisa dikumpulkan oleh player dari dunia Genshin dan tidak ada cara lain, ditambah dengan fakta bahwa bahan ini tersebar di dunia Genshin yang tidak bisa disebut kecil, karena tercatat bahwa ukuran dari dunia Genshin melebihi 81 KM^2, sehingga tanpa adanya perencanaan yang baik, player akan sangat membuang waktu hanya demi mengumpulkan bahan ini. Disini, Algoritma Dijkstra dapat membantu para player untuk mengumpulkan bahan ini dengan lebih efektif, dikarenakan algoritmanya yang dapat menghitung rute tercepat dalam pengumpulan bahan ini.

Gambar 1. Peta Dunia Genshin Impact
(sumber : reddit) diakses pada 7 Desember 2023 pukul 18.29

II. DASAR TEORI

Algoritma Dijkstra adalah salah satu algoritma pencarian jalur terpendek yang ditemukan oleh Edsger W. Dijkstra pada tahun 1956. Fokus utama algoritma ini adalah menentukan jalur terpendek dari suatu titik awal ke semua titik lainnya dalam sebuah graf berbobot. Graf yang dimaksud dapat direpresentasikan sebagai simpul-simpul yang terhubung oleh tepi atau edge, dengan setiap edge memiliki bobot atau nilai yang menunjukkan jarak antara dua simpul. Algoritma Dijkstra bekerja dengan menghitung jarak terpendek dari simpul awal ke simpul lainnya, dan secara bertahap memperbarui perkiraan jarak seiring perjalanan melalui graf.

Graf, dalam konteks algoritma Dijkstra, adalah struktur data yang terdiri dari simpul-simpul (node) yang terhubung oleh tepi (edge). Simpul mewakili entitas atau objek, sementara tepi menggambarkan hubungan atau keterkaitan antar simpul. Pada dasarnya, graf menyajikan representasi visual dari interaksi antar elemen dalam suatu sistem.

Terdapat beberapa jenis graf, dan pemahaman mengenai jenis-jenis tersebut sangat penting dalam konteks aplikasi algoritma Dijkstra. Graf dapat dibagi menjadi dua kategori utama: graf berarah dan graf tidak berarah. Graf berarah memiliki arah pada setiap tepi, yang berarti hubungan antar simpul memiliki orientasi tertentu. Sebaliknya, graf tidak berarah tidak memiliki arah pada tepi, sehingga hubungan antar simpul bersifat simetris.

Selain itu, graf dapat dibedakan menjadi graf berbobot dan graf tidak berbobot. Graf berbobot memberikan nilai numerik pada setiap tepi, yang mencerminkan properti tertentu seperti jarak atau biaya. Sebaliknya, graf tidak berbobot hanya menunjukkan adanya koneksi tanpa memperhitungkan nilai numerik.

Penerapan algoritma Dijkstra dalam permainan Genshin Impact melibatkan modelisasi lingkungan permainan sebagai graf berbobot, di mana simpul menggambarkan lokasi dalam permainan dan tepi mencerminkan jarak antar lokasi tersebut. Dengan memahami jenis-jenis graf dan bagaimana graf direpresentasikan dalam konteks ini, pemain dapat mengoptimalkan pengumpulan Starconch dengan merencanakan rute yang efisien melalui pemahaman yang lebih baik tentang struktur lingkungan permainan.

Langkah-langkah dasar algoritma Dijkstra melibatkan inisialisasi nilai jarak awal, pemilihan simpul dengan jarak terpendek, dan pembaruan jarak ke simpul-simpul tetangga. Proses ini berlanjut hingga seluruh simpul dalam graf dijelajahi. Algoritma Dijkstra menghasilkan jalur terpendek dari titik awal ke semua simpul lainnya, membentuk kerangka dasar yang dapat diadaptasi untuk mengoptimalkan rute pengumpulan Starconch dalam Genshin Impact.

Penerapan algoritma Dijkstra dalam konteks permainan ini melibatkan modelisasi lingkungan sebagai graf, dengan simpul yang mewakili lokasi dalam permainan dan tepi yang mencerminkan jarak antar lokasi tersebut. Dengan memahami prinsip-prinsip dasar ini, pemain dapat memanfaatkan algoritma Dijkstra untuk merencanakan rute pengumpulan Starconch dengan efisien dalam dunia terbuka yang dinamis.

III. Aplikasi Teori

A. Representasi Lingkungan Sebagai Graf

Dalam penerapan konsep ini ke dalam Genshin Impact, lingkungan permainan diinterpretasikan sebagai graf, di mana setiap simpul merepresentasikan lokasi potensial pengumpulan Starconch, dan tepi menggambarkan jarak atau waktu antara lokasi tersebut. Masing-masing simpul dalam graf dilengkapi dengan atribut bobot, mencerminkan seberapa besar waktu atau sumber daya yang dibutuhkan untuk mencapainya. Representasi graf ini menjadi fondasi esensial yang memungkinkan pemain untuk menerapkan algoritma Dijkstra dengan efektif.

Gambar 2. Peta Lokasi *Starconch* dalam Permainan Genshin Impact (sumber : Genshin Interactive Map) diakses pada 7 Desember 2023 pukul 20.06

B. Inisialisasi dan Perbaruan Nilai Jarak

Langkah selanjutnya dalam aplikasi teori ini melibatkan inisialisasi nilai jarak dari simpul awal ke seluruh simpul lainnya dalam graf. Proses ini berlanjut dengan pemilihan simpul yang memiliki jarak terpendek, diikuti oleh pembaruan nilai jarak ke simpul-simpul tetangga secara iteratif. Dengan kata lain, algoritma Dijkstra secara bertahap membangun jalur terpendek dari titik awal ke semua simpul lain, menghasilkan informasi kritis terkait jarak optimal yang harus ditempuh dalam konteks pengumpulan Starconch.

C. Penyesuaian Terhadap Konteks Permainan

Penting untuk dicatat bahwa penerapan algoritma Dijkstra dalam Genshin Impact tidak terlepas dari adaptasi terhadap dinamika permainan. Lingkungan dalam Genshin Impact sering mengalami perubahan, dan rute terpendek yang optimal pada suatu waktu mungkin tidak lagi relevan dalam kondisi tertentu. Oleh karena itu, pemain dapat meningkatkan aplikasi algoritma Dijkstra dengan mempertimbangkan pembaruan dinamis terhadap graf, memungkinkan refleksi yang akurat terhadap perubahan lingkungan dan menjaga keberlanjutan jalur terpendek.

D. Aplikasi Dalam Program

Dalam pengaplikasian teori ini ke dalam program, saya membaginya menjadi dua komponen utama: pertama, kode 2lgoritma Dijkstra yang terdapat dalam file 2lgoritm.py, dan kedua, kode aplikasi yang ada pada file pep.py. Inilah program dijkstranya:

```
def 2lgoritm(graph, start):
    arrayJarak = {node: float('inf') for node in graph}
    arrayJarak[start] = 0

    rute = {node: [] for node in graph}
    rute[start] = [start]

    nodes_to_visit = list(graph.keys())

    while nodes_to_visit:
```

```
        current_node = min(nodes_to_visit, key=lambda
node:arrayJarak[node])
        nodes_to_visit.remove(current_node)

        for neighbor, weight in graph[current_node].items():
            distance = arrayJarak[current_node] + weight
            if distance < arrayJarak[neighbor]:
                arrayJarak[neighbor] = distance
                rute[neighbor] = rute[current_node] + [neighbor]

    return arrayJarak, rute
```

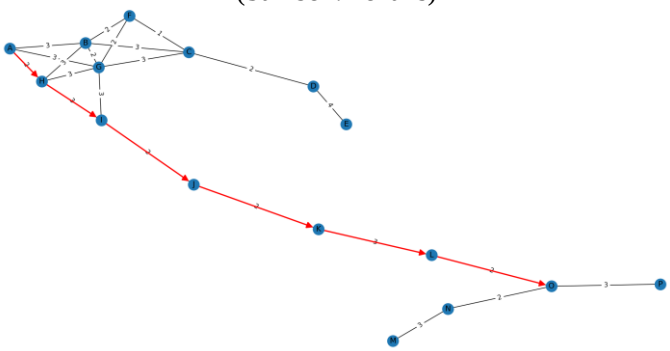
Pertama, dalam bagian kode 3lgoritma Dijkstra (3lgoritm.py), saya mengimplementasikan 3lgoritma Dijkstra untuk mencari jalur terpendek dari suatu titik awal ke semua titik lainnya dalam graf. Fungsi ini memberikan hasil jarak terpendek antara titik-titik tersebut. Selain itu, terdapat fungsi print_graph yang bertujuan untuk membantu visualisasi graf. Penggunaan library NetworkX dan Matplotlib memungkinkan kita untuk memvisualisasikan jalur terpendek yang dihasilkan oleh 3lgoritma Dijkstra, dengan memberikan warna merah pada jalur tersebut.

```
Def print_graph(graph, path):
    G = nx.Graph()

    for node, edges in graph.items():
        for edge, weight in edges.items():
            G.add_edge(node, edge, weight=weight)

    pos = nx.spring_layout(G, seed=42)
    nx.draw(G, pos, with_labels=True)
    labels = nx.get_edge_attributes(G, ‘weight’)
    nx.draw_networkx_edge_labels(G, pos,edge_labels=labels)

    H = nx.DiGraph()

    path_edges = [(path[i], path[i+1]) for I in range(len(path) –
1)]
    H.add_edges_from(path_edges)
    nx.draw_networkx_edges(H, pos, edgelist=path_edges,
edge_color=’r’, width=2, arrowstyle=’-|>’, arrowsize=20)

    plt.show()
```

Kedua, pada bagian kode aplikasi (pep.py), terdapat fungsi calculate_path yang menerima input dari pengguna berupa titik awal dan titik akhir pengumpulan Starconch. Fungsi ini kemudian menggunakan 3lgoritma Dijkstra untuk menghitung jalur terpendek dan jarak antara kedua titik tersebut. Graf yang direpresentasikan dalam kode menggambarkan keterhubungan antar titik dan bobot pada tepinya, mencerminkan struktur lingkungan Genshin Impact. Selanjutnya, program memanfaatkan input dari pengguna untuk menentukan titik awal dan akhir pengumpulan Starconch, dengan hasil berupa informasi jarak terpendek dan rute yang diambil. Berikut adalah hasil dari kedua program.

```
Masukkan titik awal pengumpulan starconch (huruf kapital): A
Masukkan titik akhir pengumpulan starconch (huruf kapital): O
Jarak terdekat dari A menuju O adalah 14

rute terdekat dari A menuju O adalah ['A', 'H', 'I', 'J', 'K', 'L', 'O']
```

Gambar 3. Output program
(Sumber : Penulis)



Gambar 4. Gambar Graf Setelah Output Program
(Sumber : Penulis)

IV. KESIMPULAN

Melalui penerapan algoritma Dijkstra dalam merencanakan rute pengumpulan Starconch dalam permainan Genshin Impact, saya dapat menyimpulkan bahwa algoritma ini memberikan kontribusi yang signifikan dalam meningkatkan efisiensi permainan. Sistematisasi jalur pengumpulan Starconch menggunakan Dijkstra memungkinkan pemain untuk menavigasi dengan lebih efektif di dalam dunia terbuka yang dinamis. Penerapan ini memberikan pemahaman yang lebih baik tentang rute terpendek dan memungkinkan pemain untuk membuat keputusan yang lebih baik seiring perjalanan mereka dalam mengumpulkan Starconch.

Visualisasi graf melalui NetworkX dan Matplotlib memberikan gambaran yang jelas tentang rute terpendek, memberikan pemain pandangan yang lebih terinci terhadap lintasan yang dihasilkan oleh algoritma Dijkstra. Penggunaan graf ini memudahkan interpretasi dan analisis rute secara visual, memperkaya pengalaman pemain dalam menjelajahi dunia terbuka permainan.

Pada sisi saran, pengembangan lebih lanjut pada aplikasi algoritma Dijkstra dalam Genshin Impact dapat mencakup integrasi faktor-faktor tambahan seperti karakteristik musuh atau dinamika cuaca yang dapat memengaruhi pengumpulan Starconch. Penelitian lebih lanjut juga dapat menggali variasi dari algoritma Dijkstra, serta pendekatan algoritma lainnya untuk permasalahan serupa. Pengembang permainan dapat mempertimbangkan pengimplementasian solusi ini dalam permainan secara resmi, memberikan pemain alat yang lebih baik untuk mengoptimalkan pengalaman mereka dalam menjelajahi dunia terbuka yang luas.

Dengan demikian, implementasi algoritma Dijkstra bukan hanya memberikan solusi efektif untuk pengumpulan Starconch, tetapi juga membuka potensi pengembangan lebih lanjut dalam konteks strategi dan pengoptimalan permainan. Dalam konteks ini, algoritma Dijkstra membuktikan diri sebagai pendekatan yang dapat meningkatkan aspek strategis dan eksploratif dari Genshin Impact.

REFERENSI

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/matdis23-24.htm
[2] Genshin Impact Interactive World Map
[3] Rinaldi Munir, Diktat Kuliah Matematika Diskrit Edisi Keempat
[4] Genshin Impact Wiki Guide & Tips
[5] Jurnal Matematika UNAND

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, 3 Desember 2023

Axel Santadi Warih 13522155