Pemecahan *Modified Travelling Salesman Problem* pada PT. XYZ dengan *Dynamic Programming*

Alvin Rizqi Alfisyahrin 13519126
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13519126@std.stei.itb.ac.id

***Abstrak*—Zaman sekarang industri berkembang dengan cepat. Setiap jenis usaha bekerja sama satu sama lain untuk mencapai target yang mereka inginkan. Perkembangan ini tentu diiringi dengan berkembangnya teknologi, setiap masalah yang ada hampir bisa diselesaikan oleh teknologi yang ada saat ini. Efisiensi merupakan salah satu faktor penting karena setiap masalah dituntut untuk dapat dipecahkan secepat mungkin dengan menggunakan *space* yang sedikit mungkin. Dalam makalah ini akan dijelaskan *Modified Travelling Salesman Problem* dengan algoritma *Dynamic Programming* untuk menentukan rute perjalanan yang paling efisien pada PT. XYZ dengan memperhatikan berbagai *costs* yang ada.**

Keywords—Travelling Salesman Problem, Modified, Dynamic Programming, Efisien, Costs, Rute, Teknologi.

I. PENDAHULUAN

PT. XYZ merupakan perusahaan yang bergerak dalam bidang konsultasi manajemen. PT. XYZ bermitra dengan para pemimpin dalam bisnis dan masyarakat untuk mengatasi permasalahan krusial dan mengambil peluang terbesar dari perusahaan mereka. Saat ini, PT. XYZ sudah bekerja sama dengan berbagai perusahaan besar yang ada di Indonesia maupun negara-negara lainnya. PT. XYZ memberikan solusi terintegrasi melalui teknologi,desain, dan konsultasi yang terdepan.

PT. XYZ menawarkan konsultasi dari berbagai bidang, baik dari segi sosial, teknologi, edukasi, *healthcare, oil and gas, travel*, dan lain lain. Tentu saja perusahaan ini tersebar dalam wilayah yang berbeda-beda. Terkadang, seorang konsultan harus mengambil beberapa proyek sekaligus dalam perjalanannya. Ada beberapa faktor yang perlu diperhatikan dalam perjalanan seorang konsultan, yaitu jarak dan biaya perjalanan dan prediksi waktu yang dihabiskan dalam kota atau negara bersangkutan.

Hal yang membuat permasalahan ini berbeda dari TSP adalah pada kasus ini ada faktor tambahan yang harus diperhatikan. TSP pada umumnya hanya memperhatikan jarak dari suatu tempat ke tempat lain. Kita perlu menentukan rute perjalanan paling efisien agar memberikan hasil yang maksimal dari perjalanan.

Dalam makalah ini penulis menggunakan penerapan salah satu strategi algoritma yaitu Algoritma A*. Penulis memilih algoritma ini karena memberikan hasil yang optimal dan memperhatikan faktor yag ada daripada algoritma *route-planning* lainnya. Penulis tidak memperhatikan faktor-faktor yang tidak terduga dalam perjalanan, seperti *delay* dalam perjalanan, pengerjaan proyek yang memakan waktu lebih lama dan lain lain.

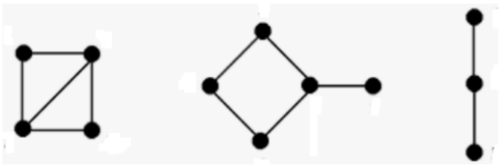
II. LANDASAN TEORI

A. *Pengertian Graf*

Graf merupakan struktur yang merepresentasikan hubungan objek-objek diskrit yang ditandai dengan simpul (vertex) dan sisi (edge). Simpul merepresentasikan objek di dalam struktur tersebut, sedangkan sisi merepresentasikan hubungan antara objek di dalam struktur tersebut. Graf dapat ditulis sebagai G = (V,E), dengan V = {v1, v2, v3, …, vn} merupakan himpunan tidak kosong dari simpul-simpulnya dan E = {e1, e2, …, en} merupakan himpunan sisi yang menghubungkan sepasang simpul di dalam graf.

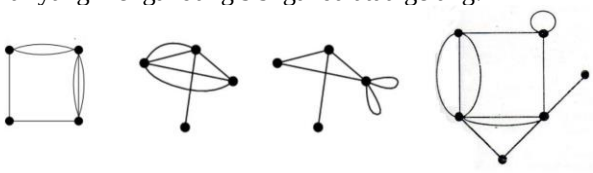
B. *Jenis Graf*

1. Graf sederhana
   Graf sederhana merupakan graf yang tidak mengandung sisi ganda maupun gelang.



Gambar 1. Graf Sederhana

2. Graf tak-sederhana
   Graf yang mengandung sisi ganda atau gelang.



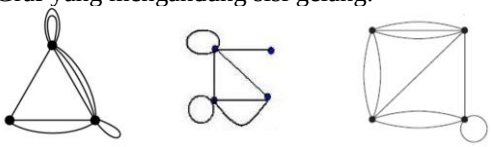
Gambar 2. Graf tak-sederhana

Graf tak-sederhana dibagi lagi menjadi dua jenis:

a. Graf ganda (*Multigraph*)
   Graf yang memiliki sisi ganda

Gambar 3. Graf Ganda

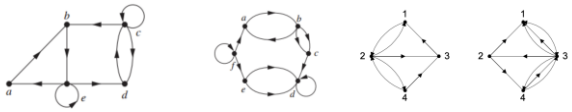
b. Graf semu (*Pseudo-graph*)
Graf yang mengandung sisi gelang.



Gambar 4. Graf semu

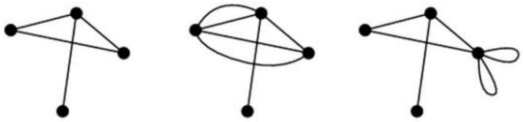
Berdasarkan ada atau tidaknya orientasi arah pada sisi, graf dapat dibagi menjadi dua jenis:

1. Graf berarah
Graf yang memiliki orientasi arah pada setiap sisinya



Gambar 4. Graf berarah

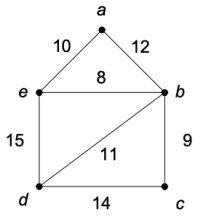
2. Graf tidak berarah
Graf yang tidak memiliki orientasi arah pada setiap sisinya



Gambar 5. Graf tidak berarah

C. *Istilah dalam Graf*

1. Graf berbobot (Weighted graph)
Graf berbobot merupakan graf yang setiap sisinya memiliki bobot tertentu.



Gambar 6. Graf berbobot

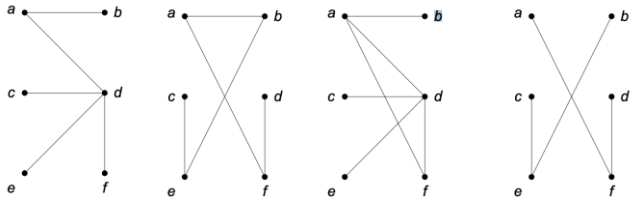
2. Lintasan (*Path*)
Lintasan dengan panjang n dari simpul awal v0 ke simpul tujuan vn di dalam graf G merupakan barisan berselang seling simpul-simpul dan sisi-sisi yang berbentuk v0, e1, v1, e2, v2, … , vn-1, en, vn sedemikian sehingga sisi-sisi dari graf G adalah e1 = (v0, v1), e2 = (v1, v2), … , en = (vn-1, vn).

3. Ketetanggan (*Adjacent*)
Dua buah simpul pada graf bertetangga jika keduanya terhubung langsung oleh sebuah sisi.

4. Sirkuit (*Circuit*)
Sirkuit merupakan lintasan yang berawal dam berakhir pada simpul yang sama.

D. *Pengertian Pohon*

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



pohon pohon bukan pohon bukan pohon

Gambar 7. Pohon

Misalkan G = (V, E) adalah graf tak-berarah sederhana dan jumlah simpulnya n. Maka, semua pernyataan di bawah ini adalah ekivalen:

1. G adalah pohon.
2. Setiap pasang simpul di dalam G terhubung dengan lintasan tunggal.
3. G terhubung dan memiliki m = n – 1 buah sisi.
4. G tidak mengandung sirkuit dan memiliki m = n – 1 buah sisi.
5. G tidak mengandung sirkuit dan penambahan satu sisi pada graf akan membuat hanya satu sirkuit.
6. G terhubung dan semua sisinya adalah jembatan

E. *Modified Travelling Salesman Problem*

Travelling Salesman Problem merupakan salah satu permasalahan sirkuit Hamilton, yaitu untuk mencari biaya minimum tur dari beberapa kota yang terhubung jika mengunjungi setiap kota tepat satu kali. Jika terdapat himpunan tempat yang membentuk graf terhubung, maka jalur yang mengubungkan tempat-tempat tersebut mempunyai biaya atau harga tertentu yang dapat direpresentasikan dengan graf berbobot. Untuk suatu graf lengkap yang terdiri dari n kota, terdapat $\frac{(n-1)!}{2}$! sirkuit Hamilton, yang berarti terdapat $\frac{(n-1)!}{2}$! rute yang berbeda untuk mengunjungi setiap kota dan kembali ke tempat asal. Permasalahan TSP tidak terbatas hanya pada graf lengkap asalkan graf tersebut memiliki sirkuit Hamilton.

Untuk kasus *Modified*, permasalahannya tidak hanya dari jarak, akan tetapi juga waktu, biaya, dan faktor lainnya sehingga permasalahan menjadi lebih kompleks.

F. *Lintasan dan Sirkuit Hamilton*

Lintasan Hamilton merupakan lintasan yang melalui setiap simpul di dalam graf tepat satu kali. Sirkuit Hamilton merupakan sirkut yang melalui setiap simpul di dalam graf tepat satu kali, kecuali simpul asal yang sekaligus sebagai simpul akhir yang dilalui dua kali. Graf dapat dikatakan sebagai graf Hamilton apabila memiliki sirkuit Hamilton. Jika graf hanya

memiliki lintasan Hamilton, maka graf tersebut dapat dikatakan sebagai graf semi-Hamilton.

G. *Dynamic Programming* (Pemrograman Dinamis)

Pemrograman Dinamis terutama merupakan pengoptimalan atas rekursi biasa. Di mana pun kami melihat solusi rekursif yang memiliki panggilan berulang untuk input yang sama, kami dapat mengoptimalkannya menggunakan Pemrograman Dinamis. Idenya adalah untuk hanya menyimpan hasil submasalah, sehingga kita tidak perlu menghitung ulang jika diperlukan nanti. Pengoptimalan sederhana ini mengurangi kerumitan waktu dari eksponensial menjadi polinomial. Misalnya, jika kita menulis solusi rekursif sederhana untuk Bilangan Fibonacci, kita mendapatkan kompleksitas waktu eksponensial dan jika kita mengoptimalkannya dengan menyimpan solusi sub-masalah, kompleksitas waktu berkurang menjadi linier.

Metode ini dikembangkan oleh Richard Bellman pada 1950-an dan telah digunakan di berbagai bidang, mulai dari teknik kedirgantaraan hingga ekonomi. Ada 2 jenis pendekatan dalam Pemrograman Dinamis, yaitu:

1. Pendekatan top-down
   Ini adalah hasil langsung dari formulasi rekursif dari masalah apa pun. Jika solusi untuk masalah apa pun dapat dirumuskan secara rekursif menggunakan solusi untuk sub-masalah, dan jika sub-masalah tersebut tumpang tindih, maka seseorang dapat dengan mudah memoisasi atau menyimpan solusi untuk sub-masalah dalam sebuah tabel. Setiap kali kita mencoba untuk memecahkan sub-masalah baru, pertama-tama kita memeriksa tabel untuk melihat apakah sudah terpecahkan. Jika solusi telah dilaporkan, kita dapat secara langsung, jika tidak menyelesaikan sub-masalah dan menambahkan solusinya ke tabel.
2. Pendekatan bottom-up
   Setelah kita merumuskan solusi untuk suatu masalah secara rekursif dalam sub-masalah, kita dapat mencoba merumuskan kembali masalah secara bottom-up: coba selesaikan sub-masalah terlebih dahulu dan gunakan solusi mereka untuk membangun dan sampai pada solusi untuk sub -masalah yang lebih besar. Ini juga biasanya dilakukan dalam bentuk tabel dengan menghasilkan solusi secara berulang untuk sub-masalah yang lebih besar dan lebih besar dengan menggunakan solusi untuk sub-masalah kecil

Karena Pemrograman Dinamis memanggil fungsi rekursif, maka penentuan basis dan rekurens untuk kasus *Travelling Salesman Problem* adalah:

Hubungan rekursif:

$$f(1, V-\{1\}) = \min_{2 \le k \le n}\{c_{1k} + f(k, V-\{1,k\})\} \quad (1)$$

Dengan merampatkan persamaan (1), diperoleh

$$f(i, \varnothing) = c_{i,1} \quad , \ 2 \le i \le n \quad \text{(basis)}$$
$$f(i, S) = \min_{j \in S}\{c_{ij} + f(j, S-\{j\})\} \quad \text{(rekurens)} \ (2)$$

Gambar 8. Rekurens dan Basis

Berikutnya merupakan *pseudocode* penyelesaian *Travelling Salesman Problem* menggunakan Pemrograman Dinamis:

```
Algorithm 1: Dynamic Approach for TSP
Data: s: starting point; N: a subset of input cities; dist():
      distance among the cities
Result: Cost : TSP result
Visited[N] = 0;
Cost = 0;
Procedure TSP(N, s)
    Visited[s] = 1;
    if |N| = 2 and k ≠ s then
        Cost(N, k) =dist(s, k);
        Return Cost;
    else
        for j ∈ N do
            for i ∈ N and visited[i] = 0 do
                if j ≠ i and j ≠ s then
                    Cost(N, j) = min ( TSP(N − {i}, j) + dist(j, i))
                    Visited[j] = 1;
                end
            end
        end
    end
    Return Cost;
end
```

Gambar 9. *Pseudocode* TSP dengan Pemrograman Dinamis

Langkah-langkah pengembangan Algoritma Pemrograman Dinamis adalah:

1. Karakteristikkan struktur solusi optimal.
   Berupa tahap, variable keputusan, status (state), dsb
2. Definisikan secara rekursif nilai solusi optimal.
   Hubungan nilai optimal suatu tahap dengan tahap sebelumnya
3. Hitung nilai solusi optimal secara maju atau mundur.
   Menggunakan tabel
4. Rekonstruksi solusi optimal (opsional).
   Rekonstruksi solusi secara mundur

H. *Fungsi Heuristik*

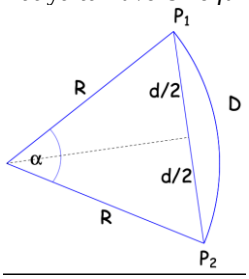
Teknik heuristik, atau heuristik (/ hjʊəˈrɪstɪk /; Yunani Kuno: εὑρίσκω, heurískō, 'Saya menemukan, menemukan'), adalah pendekatan untuk pemecahan masalah atau penemuan jalan yang menggunakan metode praktis yang tidak dijamin akan optimal, sempurna, atau rasional, namun tetap cukup untuk mencapai tujuan atau perkiraan jangka pendek dan langsung. Jika menemukan solusi optimal tidak mungkin atau tidak praktis, metode heuristik dapat digunakan untuk mempercepat proses menemukan solusi yang optimal.

Fungsi heuristik dikatakan *admissible* jika tidak pernah melebihi biaya untuk mencapai tujuan, yaitu biaya yang

diperkirakan untuk mencapai tujuan tidak lebih tinggi dari biaya serendah mungkin dari titik saat ini di jalur.

I. Straight-line Distance

Straight-line Distance (SLD) pada Bumi merupakan jarak garis lurus antara 2 titik koordinat yang ada di Bumi. Penulis menggunakan *straight-line distance* sebagai fungsi heuristic karena jarak ini selalu tidak pernah melebihi jarak sebenarnya dari kedua titik koordinat yaitu *haversine function*.



Gambar 10. Busur Lingkaran

α merupakan perbedaan sudut antara 2 titik yang ada di permukaan Bumi, dimana dalam kasus ini menggunakan *Teorema Phytagoras*

$$\alpha = \sqrt{(P_{1,lat} - P_{2,lat)}{}^2 + (P_{1,long} - P_{2,long)}{}^2}$$

Lalu jarak garis lurus antara kedua titik tersebut dapat dihitung dengan

$$d = 2R \sin(\frac{\alpha}{2})$$

Dimana R merupakan Radius Bumi yang bernilai 6371 km.

III. METODOLOGI DAN HASIL PENELITIAN

A. Pendefinisian Kasus

Penulis mengambil kasus seorang konsultan yang sedang berada di Jakarta, Indonesia dengan titik alamat beserta titik koordinat (Lintang, Bujur):

0. Sampoerna Strategic Square, 19th Floor. Jl. Jenderal Sudirman Kav. 45-46 ,Jakarta, Indonesia (-6.2167, 106.8183)

Berikut merupakan lokasi proyek yang akan ditugaskan kepada konsultan tersebut:

1. Google
   1600 Amphitheatre Parkway, Mountain View, California, Amerika Serikat
   (37.42229705405835, -122.08407294474938)
2. Saudi Arabian Oil Company (*ID Office*)
   6767 طريق الامير سلطان بن عبدالعزيز، Najmah, 2964 Ras Tanura 32814 2964, Arab Saudi
   (27.130917238192765, 50.06032792955813)
3. China Construction Bank
   Lancang Lahu Autonomous County, Pu'er City, Yunnan, Tiongkok
   (22.55377503461339, 99.93113612812857)
4. Nestle
   Vevey, Vaud, Switzerland
   (46.4672981627121, 6.8461285401625895)

Konsultan menggunakan penerbangan udara dalam perjalanan ini, berikut merupakan rute pesawat yang ada pada maskapai KLM yang berkolaborasi dengan PT. XYZ:

1. Indonesia – China
   SLD = 3285.752 km
2. Indonesia – Amerika Serikat
   SLD = 13986.957 km
3. Indonesia – Saudi Arabia
   SLD = 7136.048 km
4. Indonesia – Switzerland
   SLD = 11271.414 km
5. Amerika Serikat – Saudi Arabia
   SLD = 12790.380 km
6. Amerika Serikat – China
   SLD = 12027.996 km
7. Amerika Serikat – Switzerland
   SLD = 9389.626 km
8. Saudi Arabia – China
   SLD = 5026.0 km
9. Saudi Arabia – Switzerland
   SLD = 4334.538 km
10. China – Switzerland
    SLD = 8438.256 km

Berikut merupakan perkiraan minimal waktu yang dihabiskan oleh konsultan tersebut di setiap kota yang ia hampiri:

1. Google (Amerika Serikat) = 1 minggu
2. Saudi Arabia Oil Company (Saudi Arabia) = 2 minggu
3. China Construction Bank (China) = 4 minggu
4. Nestle (Switzerland) = 3 minggu

B. Batasan dan Asumsi Penelitian

Dalam penulisan makalah ini, terdapat beberapa batasan dan asumsi yang digunakan, yaitu:

1. Biaya perjalanan sebanding dengan jarak. Hal ini berarti jika jarak semakin jauh, maka biaya yang dibutuhkan juga semakin banyak dan begitu juga sebaliknya.
2. Setiap perusahaan proyek bisa dikunjungi kapan saja berdasarkan hasil penentuan rute ini.
3. Waktu dalam perjalanan sebanding dengan jarak, Hal ini berarti jika jarak semakin jauh, maka waktu yang dibutuhkan juga lebih banyak.
4. Perjalanan dimulai dari Indonesia dan berakhir di Indonesia.
5. Waktu yang dihabiskan di setiap negara merupakan waktu minimum tanpa ada faktor apapun.

C. Penentuan Cost dan Matriks

Cost dapat ditentukan dengan cara seperti berikut:

$$cost_{ij} = \frac{SLD_{ij}}{1000} + waktu_j$$

Penulis memilih persamaan ini agar faktor jarak SLD dan waktu yang dihabiskan di setiap negara memiliki orde yang sama, yaitu puluhan. Pada kasus ini setiap negara ordenya adalah 10^3, sedangkan waktu dalam orde 10^1.

Nilai cost untuk setiap negara dalam bentuk matriks adalah:

$$\begin{bmatrix} 0 & 14.986 & 9.136 & 7.285 & 14.271 \\ 13.986 & 0 & 14.790 & 16.027 & 12.389 \\ 7.136 & 13.790 & 0 & 9.026 & 7.334 \\ 3.285 & 13.027 & 7.026 & 0 & 11.438 \\ 11.271 & 10.389 & 6.334 & 12.438 & 0 \end{bmatrix}$$

D. *Penyelesaian masalah menggunakan Dynamic Programming*

Masalah ini memiliki beberapa tahap dalam penyelesaiannya, seperti pada *nature* dari Pemrograman Dinamis.

- Tahap 1

$$f(i,\emptyset) = c_{i1}, \qquad 2 \le i \le n$$

Diperoleh hasil:

$$f(2,\emptyset) = c_{21} = 13.986$$
$$f(3,\emptyset) = c_{31} = 7.136$$
$$f(4,\emptyset) = c_{41} = 3.285$$
$$f(5,\emptyset) = c_{51} = 11.271$$

- Tahap 2

$$f(i,S) = \min\{c_{ij} + f(j,(S-\{j\}))\}, |S| = 1$$

Diperoleh hasil:

$$f(2,\{3\}) = \min\{c_{23} + f(3,\emptyset)\} = \min\{14.79 + 7.136\} = 21.926$$
$$ff(2,\{4\}) = \min\{c_{24} + f(4,\emptyset)\} = \min\{16.027 + 3.285\} = 19.312$$
$$f(2,\{5\}) = \min\{c_{25} + f(5,\emptyset)\} = \min\{12.389 + 11.271\} = 23.660$$
$$f(3,\{2\}) = \min\{c_{32} + f(2,\emptyset)\} = \min\{13.79 + 13.986\} = 27.776$$
$$f(3,\{4\}) = \min\{c_{34} + f(4,\emptyset)\} = \min\{9.026 + 3.285\} = 12.311$$
$$f(3,\{5\}) = \min\{c_{35} + f(5,\emptyset)\} = \min\{7.334 + 11.271\} = 18.605$$
$$f(4,\{2\}) = \min\{c_{42} + f(2,\emptyset)\} = \min\{13.027 + 13.986\} = 27.013$$
$$f(4,\{3\}) = \min\{c_{43} + f(3,\emptyset)\} = \min\{7.026 + 7.136\} = 14.162$$
$$f(4,\{5\}) = \min\{c_{45} + f(5,\emptyset)\} = \min\{11.438 + 11.271\} = 22.709$$
$$f(5,\{2\}) = \min\{c_{52} + f(2,\emptyset)\} = \min\{10.389 + 13.986\} = 24.375$$
$$f(5,\{3\}) = \min\{c_{53} + f(3,\emptyset)\} = \min\{6.334 + 7.136\} = 13.470$$
$$f(5,\{4\}) = \min\{c_{54} + f(4,\emptyset)\} = \min\{12.438 + 3.285\} = 15.723$$

- Tahap 3

$$f(i,S) = \min\{c_{ij} + f(j,(S-\{j\}))\}, |S| = 2 \; dan \; i \neq 1, 1 \notin S \; dan \; i \notin S$$

Diperoleh hasil:

$$f(2,\{3,4\}) = \min\{c_{23} + f(3,\{4\}), c_{24} + f(4,\{3\})\} = \min\{14.790 + 12.311, 16.027 + 14.162\} = 27.101$$
$$f(2,\{3,5\}) = \min\{c_{23} + f(3,\{5\}), c_{25} + f(5,\{3\})\} = \min\{14.790 + 18.605, 12.389 + 13.470\} = 25.859$$
$$f(2,\{4,5\}) = \min\{c_{24} + f(4,\{5\}), c_{25} + f(5,\{4\})\} = \min\{16.027 + 22.709, 12.389 + 15.723\} = 28.112$$
$$f(3,\{2,4\}) = \min\{c_{32} + f(2,\{4\}), c_{34} + f(4,\{2\})\} = \min\{13.790 + 19.312, 9.026 + 27.013\} = 33.102$$
$$f(3,\{2,5\}) = \min\{c_{32} + f(2,\{5\}), c_{35} + f(5,\{2\})\} = \min\{13.790 + 23.660, 7.334 + 24.375\} = 31.709$$
$$f(3,\{4,5\}) = \min\{c_{34} + f(4,\{5\}), c_{35} + f(5,\{4\})\} = \min\{9.026 + 22.709, 7.334 + 15.723\} = 23.057$$
$$f(4,\{2,3\}) = \min\{c_{42} + f(2,\{3\}), c_{43} + f(3,\{2\})\} = \min\{13.027 + 21.926, 7.026 + 27.776\} = 34.802$$
$$f(4,\{2,5\}) = \min\{c_{42} + f(2,\{5\}), c_{45} + f(5,\{2\})\} = \min\{13.027 + 23.660, 11.438 + 24.375\} = 36.687$$

$$f(4,\{3,5\}) = \min\{c_{43} + f(3,\{5\}), c_{45} + f(5,\{3\})\} = \min\{7.026 + 18.605, 11.438 + 13.470\} = 25.631$$

$$f(5,\{2,3\}) = \min\{c_{52} + f(2,\{3\}), c_{53} + f(3,\{2\})\} = \min\{10.389 + 21.926, 6.334 + 27.776\} = 34.110$$

$$f(5,\{2,4\}) = \min\{c_{52} + f(2,\{4\}), c_{54} + f(4,\{2\})\} = \min\{10.389 + 19.312, 12.438 + 27.013\} = 29.701$$

$$f(5,\{3,4\}) = \min\{c_{53} + f(3,\{4\}), c_{54} + f(4,\{3\})\} = \min\{6.334 + 12.311, 12.438 + 14.162\} = 15.645$$

- Tahap 4

$$f(i,S) = \min\{c_{ij} + f(j,(S-\{j\}))\}, |S| = 3 \; dan \; i \neq 1, 1 \notin S \; dan \; i \notin S$$

Diperoleh hasil:

$$f(2,\{3,4,5\}) = \min\{c_{23} + f(3,\{4,5\}), c_{24} + f(4,\{3,5\}), c_{25} + f(5,\{3,4\})\} = \min\{14.790 + 23.057, 16.027 + 25.631, 12.389 + 15.645\} = 28.034$$

$$f(3,\{2,4,5\}) = \min\{c_{32} + f(2,\{4,5\}), c_{34} + f(4,\{2,5\}), c_{35} + f(5,\{2,4\})\} = \min\{13.790 + 28.112, 9.026 + 36.687, 7.334 + 29.701\} = 37.035$$

$$f(4,\{2,3,5\}) = \min\{c_{42} + f(2,\{3,5\}), c_{43} + f(3,\{2,5\}), c_{45} + f(5,\{2,3\})\} = \min\{13.027 + 25.859, 7.026 + 31.709, 11.438 + 34.110\} = 38.735$$

$$f(5,\{2,3,4\}) = \min\{c_{52} + f(2,\{3,4\}), c_{53} + f(3,\{2,4\}), c_{54} + f(4,\{2,3\})\} = \min\{10.389 + 27.101, 6.334 + 33.102, 12.438 + 34.802\} = 37.490$$

- Tahap 5

$$f(1,\{2,3,4,5\}) = \min\{c_{12} + f(2,\{3,4,5\}), c_{13} + f(3,\{2,4,5\}), c_{14} + f(4,\{2,3,5\}), c_{15} + f(5,\{2,3,4\})\} = \min\{14.986 + 28.034, 9.136 + 37.035, 7.285 + 38.735, 14.271 + 37.490\} = 43.020$$

Maka, *cost minimum* yang terbentuk dari permasalahan ini adalah 43.020 . Selanjutnya, kita akan menentukan rute perjalanan dengan *minimum cost* ini. Pertama, kita meninjau pada setiap f(i,S) nilai j yang meminimumkan persamaan.

- Penentuan Rute Perjalanan

Untuk 1,{2,3,4,5} nilai j yang meminimkannya adalah 2. Maka perjalanan dimulai dari node 1 ke node 2. Selanjutnya, pada tahap 4 kita tinjau 2,{3,4,5}, nilai j yang meminimkannya adalah 5. Lalu, kita tinjau tahap 3 semua node 5, yaitu (5,{2,3}), (5,{3,4}), dan (5,{2,4}). Didapat nilai j yang meminimkannya adalah 4. Karena tersisa 1 node yang belum dikunjungi, maka node selanjutnya adalah 3, lalu kembali ke titik asal yaitu 1.

E. Hasil Penelitian

Berdasarkan hasil pencarian rute yang memiliki *cost* minimum menggunakan Algoritma Pemrograman Dinamis. Rute yang didapat adalah PT. XYZ (Indonesia) – Google (Amerika Serikat) – Nestle (Switzerland) – China Construction Bank (China) – Saudi Arabian Oil Company (Saudi Arabia) dengan total jarak tempuh 43976.887 km. Penulis menggunakan asumsi biaya perjalanan dan waktu perjalanan sebanding dengan jarak. Maka, solusi ini merupakan yang paling efisien dari segi jarak, biaya, waktu perjalanan, dan waktu yang dihabiskan di setiap negara.

IV. KESIMPULAN

Graf dapat diaplikasikan untuk menyelesaikan banyak permasalahan, salah satunya adalah untuk mencari rute terpendek ketika mengunjungi beberapa tempat tepat satu kali hingga kembali ke tempat awal atau yang biasa diebut dengan Travelling Salesman Problem (TSP). Salah satu cara untuk menyelesaikan permasalahan TSP adalah dengan menggunakan Pemrograman Dinamis

Penerapan Travelling Salesman Problem dengan algoritma Pemrograman Dinamis untuk mencari rute perjalanan dengan *cost* paling minimum baik dari segi jarak, biaya, dan waktu dari sebuah perusahaan konsultasi manajemen yang bernama PT. XYZ dalam proyek konsultasi yang memakan waktu berbulan-bulan.

Dengan mengoptimasi rute perjalanan, maka waktu dan biaya yang diperlukan dalam perjalanan juga akan menjadi lebih singkat. Pemrograman Dinamis merupakan salah satu algoritma yang sering memberikan hasil yang maksimal dan algoritma ini berjalan dengan cepat.

V. UCAPAN TERIMA KASIH

Penulis mengucapkan puji syukur kepada Tuhan Yang Maha Esa karena atas berkat dan rahmat-Nya, penulis bisa menyelesaikan tugas makalah ini. Penulis juga mengucapkan terimakasih kepada Pak Dwi Hendratmo Widyantoro selaku dosen mata kuliah IF2211 Strategi Algoritma, yang selama ini mengajar penulis dan juga memberikan ilmu sehingga saya bisa mengerjakan makalah ini. Penulis juga mengucapkan terima kasih kepada keluarga dan teman yang telah membantu saya dalam pengerjaan makalah ini. Penulis berterima kasih kepada

Bapak Rinaldi Munir yang telah menyediakan website yang sangat membantu penulis dalam pembuatan makalah ini.

REFERENSI

[1] Munir, Rinaldi. Graf Bagian 1. Informatika, Bandung: 2020. Diakses pada 10 Mei 2021.
[2] Munir, Rinaldi. Graf Bagian 2. Informatika, Bandung: 2020. Diakses pada 10 Mei 2021.
[3] Munir, Rinaldi. Graf Bagian 3. Informatika, Bandung: 2020. Diakses pada 10 Mei 2021
[4] Munir, Rinaldi. Pemrograman Dinamis Bagian 1. Informatika, Bandung: 2020. Diakses pada 11 Mei 2021
[5] Munir, Rinaldi. Pemrograman Dinamis Bagian 2. Informatika, Bandung: 2020. Diakses pada 11 Mei 2021
[6] https://www.geeksforgeeks.org/travelling-salesman-problem-set-1/
[7] https://www.baeldung.com/cs/tsp-dynamic-programming

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, 11 Mei 2021

Alvin Rizqi Alfisyahrin