Penerapan Algoritma *Held-Karp* di Pusat Belanja dalam Meminimalisir Waktu dan Kontak Fisik Konsumer selama Pandemi

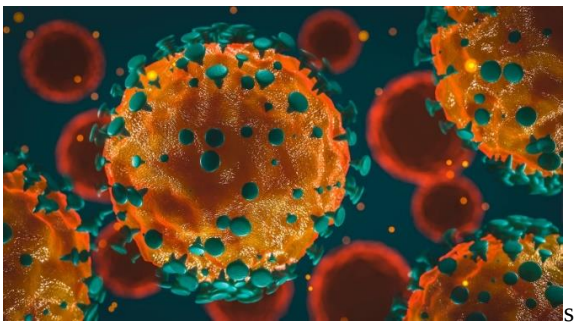
I Gede Arya Raditya Parameswara - 13520036
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13520036@std.stei.itb.ac.id

***Abstrak*—Pandemi merupakan sebuah epidemi yang menyebar ke berbagai negara di dunia dan umumnya menyerang banyak orang. Bencana ini sangat merugikan banyak pihak, salah satunya pusat belanja yang terpaksa tutup karena kekhawatiran terjadinya penyebaran virus pada lokasi tersebut. Cara yang paling tepat agar pusat belanja tetap buka disaat pandemi ini adalah dengan meminimalisir terjadinya kontak fisik dan waktu belanja konsumer. Hal ini bisa dibantu dengan penggunaan graf berbobot untuk mengarahkan dan memberikan rute kepada konsumer menuju toko-toko target konsumer berbelanja. Pada graf berbobot tersebut akan diterapkan pemrograman dinamis dan algoritma *Held-Karp* untuk mendapatkan rute terbaik yang diperlukan.**

***Kata kunci*— Held-Karp, Traveling Salesman Problem, Pusat Belanja, Pandemi**

I. PENDAHULUAN

Masa pandemi telah merubah segala peradaban kehidupan sosial serta cara beraktivitas manusia sehari-hari di masyarakat. Pada masa ini juga pemerintah mengubah berbagai aturan untuk mengetatkan serta mencegah penyebaran dan penularan dari virus pada pandemi tersebut. Usaha pemerintah dalam mengamankan sektor perdagangan di pusat belanja untuk melindungi generasi bangsa dari paparan dan penularan virus menular Covid-19 adalah mengeluarkan kebijakan untuk membatasi pengunjung pada pusat belanja serta pengunjung diwajibkan melakukan pemindaian *QR Code* sebelum memasukinya. Kebijakan ini tidak begitu efektif karena masih besar kemungkinannya untuk terjadi kontak fisik satu sama lain.



***Gambar 1.1** Ilustrasi Virus Covid-19*
(sumber: https://www.gard.no/web/updates/content /29797406/an-introduction-to-testing-for-covid-19-)

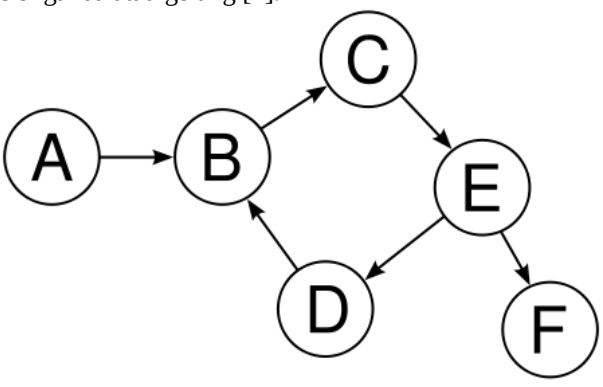
Ketika para konsumer memasuki pusat belanja, biasanya mereka akan bingung dengan denah pusat belanja tersebut jika baru pertama kali memasukinya. Hal ini mengakibatkan waktu konsumer pada pusat belanja terbuang lebih lama dan memiliki kemungkinan terjadinya kontak fisik yang lebih tinggi. Seharusnya pemerintah mengembangkan fitur pada aplikasinya, Peduli Lindungi, untuk mengurangi waktu konsumer ketika berada di pusat belanja. Fitur tersebut bisa lebih mudah dikembangkan jika digambarkan dengan pendekatan graf berbobot dan algoritma *Held-Karp*.

Graf digunakan untuk merepresentasikan objek-objek diskrit dan hubungan antara objek-objek tersebut [1]. Secara sederhana graf didefinisikan sebagai kumpulan titik yang memiliki nilai dan dihubungkan dengan garis satu sama lain.

II. LANDASAN TEORI

A. *Graf*

Graf adalah himpunan vertex (V) yang elemennya disebut simpul dan himpunan sisi (E) yang menghubungkan simpul. Berdasarkan ada atau tidaknya sisi ganda atau gelang, graf dibagi menjadi dua jenis, yaitu graf sederhana dan graf tak sederhana. Graf sederhana adalah graf yang tidak memiliki sisi ganda, sedangkan graf tak sederhana adalah graf yang memiliki sisi ganda atau gelang [2].=

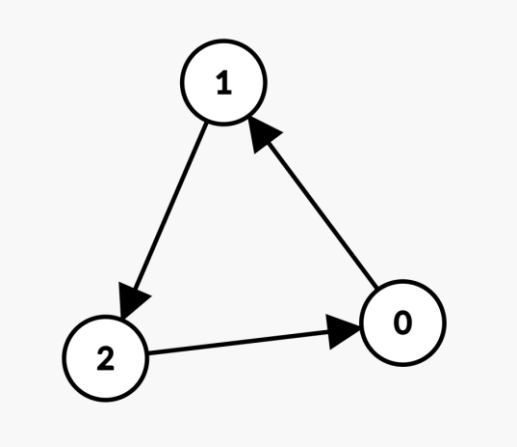


***Gambar 2.1** Graf*
(Sumber: https://commons.wikimedia.org/ wiki/File:Directed_graph,_cyclic.svg)

Berdasarkan arahnya, graf dapat dibagi menjadi dua jenis, yaitu graf berarah dan graf tak berarah. Selain berdasarkan

arahnya, graf juga dapat dibagi menjadi dua berdasarkan bobotnya, yaitu graf berbobot dan graf tak-berbobot.

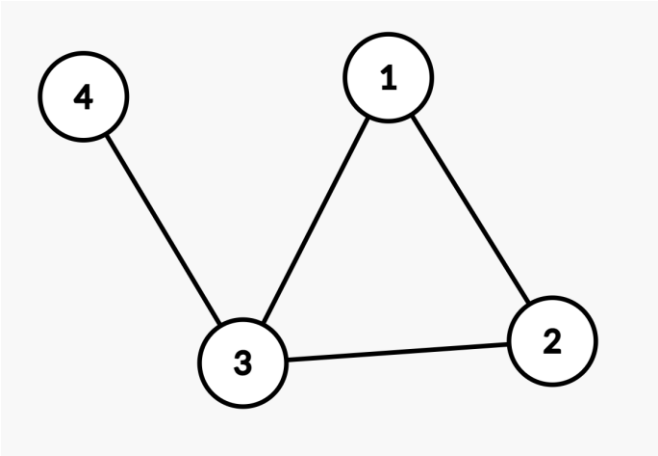
1. Graf Berarah



***Gambar 2.2** Graf Berarah*
(Sumber: Dokumen Penulis)

Graf berarah, disebut juga digraf, adalah graf yang sisi-sisinya memiliki arah. Ini biasanya ditunjukkan dengan panah di tepi; Secara lebih formal, jika v dan w adalah simpul, maka sisi adalah pasangan tak beraturan {v,w}, sedangkan sisi berarah, disebut busur, adalah pasangan terurut (v,w) atau (w,v).

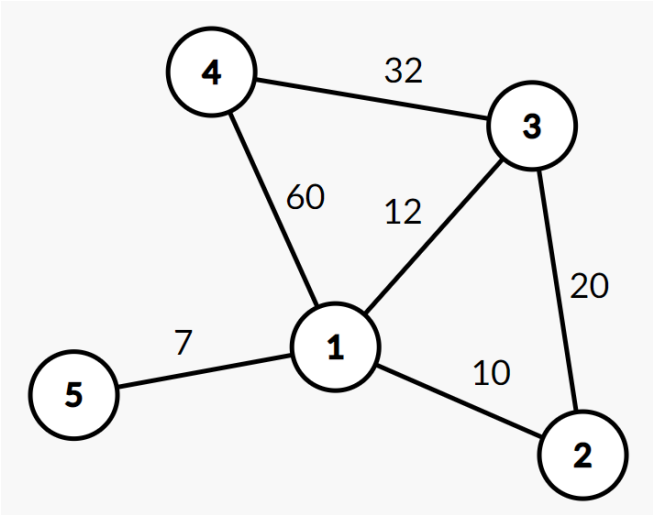
2. Graf Tak Berarah



***Gambar 2.3** Graf Tak berarah*
(Sumber: Dokumen Penulis)

Graf tak berarah adalah himpunan titik dan himpunan sisi antar titik. Setiap titik disebut vertex, setiap sisi disebut edge, dan setiap edge menghubungkan dua vertex. Urutan dua simpul yang terhubung tidak penting. Graf tak berarah adalah himpunan berhingga dari simpul-simpul bersama-sama dengan himpunan berhingga [3].

3. Graf Berbobot



***Gambar 2.4** Graf Berbobot*
(Sumber: Dokumen Penulis)

Graf berbobot adalah graf yang setiap cabangnya diberi bobot numerik. Oleh karena itu, graf berbobot adalah jenis khusus dari graf berlabel di mana labelnya adalah angka (yang biasanya dianggap positif). Terkadang, ∞ juga diperbolehkan sebagai bobotnya, namun hanya dalam kasus-kasus tertentu saja atau ketika diperlukan untuk melakukan optimisasi masalah.

B. Pemrograman Dinamis

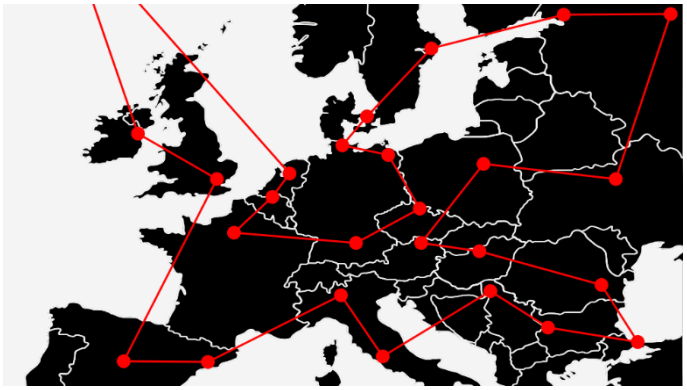
		G	A	A	T	T	C	A	G	T	T	A
	0	0	0	0	0	0	0	0	0	0	0	0
G	0	1	1	1	1	1	1	1	1	1	1	1
G	0	1	1	1	1	1	1	1	2	2	2	2
A	0	1	1	2	2	2	2	2	2	2	2	3
T	0	1	2	2	3	3	3	3	3	3	3	3
C	0	1	2	2	3	3	4	4	4	4	4	4
G	0	1	2	2	3	3	4	4	5	5	5	5
A	0	1	2	3	3	3	4	5	5	5	5	6

***Tabel 2.1** Pemrograman Dinamis*
(Sumber: https://kraulis.se/lectures/molbioinfo2001/dynprog/dynamic.html)

Pemrograman dinamis utamanya merupakan optimisasi dari sebuah rekursi biasa. Ketika melihat solusi rekursif yang selalu memanggil indeks berulang untuk input yang sama, maka pasti dapat dioptimalkan dengan pemrograman dinamis. Idenya adalah hanya dengan menyimpan hasil dari submasalah, sehingga tidak perlu menghitung ulang ketika dibutuhkan lagi nanti. Optimisasi sederhana ini mengurangi kompleksitas waktu dari eksponensial ke polynomial. Misalnya, ketika menulis solusi rekursif sederhana untuk bilangan Fibonacci, kompleksitas yang didapatkan adalah eksponensial. Namun,

jika dioptimalkan dengan menyimpan solusi submasalah, kompleksitas waktu akan berkurang menjadi linier.

C. Traveling Salesman Problem (TSP)



***Gambar 2.5** Ilustrasi Traveling Salesman Problem*
(Sumber: https://towardsdatascience.com/animating-the-traveling-salesman-problem-56da20b95b2f)

Traveling Salesman Problem (TSP) pertama kali dirumuskan pada tahun 1930 oleh Karl Menger dan sejak itu menjadi salah satu yang paling mempelajari masalah dalam optimasi. Masalahnya dijelaskan sebagai berikut: diberikan satu set kota, jarak antara setiap dua kota, dan satu penjual. Jalur optimal pelayaran dihitung, lalu diikuti oleh penjual untuk mengunjungi setiap kota tepat sekali dan kembali ke kota awal, sambil mempertahankan biaya perjalanan seminimal mungkin. Aplikasi untuk TSP jauh melampaui tur penjual sederhana, aplikasinya bisa digunakan untuk perutean kendaraan, logistik, pencetakan, dan penyolderan papan PCB. TSP dianggap Non-Deterministic Polynomial-time hard (NP-hard). Untuk menyelesaikannya, ada solusi eksak dan aproksimasi. Solusi eksak TSP mencoba semua permutasi dari jumlah titik dengan kompleksitas O(N!). Sehingga sangatlah tidak efisien, bahkan untuk angka yang kecil seperti N=10, sudah menjalankan program lebih dari 3628800, yang jelas tidak layak untuk di praktikan [6].

D. Algoritma Held-Karp

Algoritma Held–Karp, juga disebut algoritma Bellman–Held–Karp, adalah algoritma pemrograman dinamis yang diusulkan pada tahun 1962 secara independen oleh Bellman dan oleh Held dan Karp untuk memecahkan masalah traveling salesman (TSP), di mana inputnya adalah matriks jarak antara sekumpulan kota, dan tujuannya adalah untuk menemukan tur berdurasi minimum yang mengunjungi setiap kota tepat satu kali sebelum kembali ke titik awal. Ia menemukan solusi yang tepat untuk masalah ini, dan untuk beberapa masalah terkait termasuk masalah siklus Hamilton, dalam waktu eksponensial.

Algoritma ini memiliki kompleksitas yang jauh lebih baik daripada kompleksitas solusi eksak TSP. Algoritma ini memiliki kompleksitas $O(2^N N^2)$ sedangkan algoritma solusi eksak TSP adalah O(N!)..

```
function algorithm TSP (G, n) is
    for k := 2 to n do
        C({k}, k) := d1,k
    end for

    for s := 2 to n−1 do
        for all S ⊆ {2, ..., n}, |S| =
s do
            for all k ∈ S do
                C(S, k) := minm≠k,m∈S
[C(S\{k}, m) + dm,k]
            end for
        end for
    end for

    opt := mink≠1 [C({2, 3, ..., n}, k)
+ dk, 1]
    return (opt)
end function
```

***Gambar 2.6** Pseudocode Held-Karp Algorithm*
(Sumber: https://en.wikipedia.org/wiki/Held–Karp_algorithm)

E. Pusat Belanja

Pengertian dari pusat perbelanjaan adalah kompleks toko ritel dan fasilitas yang direncanakan sebagai kelompok terpadu untuk memberikan kenyamanan berbelanja yang maksimal kepada pelanggan dan penataan barang dagangan yang terekspos secara maksimal [7].

***Gambar 2.7** Ilustrasi Pusat Belanja*
(Sumber: https://www.klepierre.com/en/our-malls/porta-di-roma)

Aktivitas yang dilakukan oleh para pengunjung dalam pusat perbelanjaan sangat beragam. Mulai dari berbelanja, menikmati makanan di restoran favorit atau sekedar berjalan-jalan dan sekedar melihat-lihat saja (*window-shopping*), semua dapat dinikmati di pusat perbelanjaan tersebut. Disamping fungsi utama sebagai tempat berbelanja, pusat perbelanjaan pada umumnya menyedikan sarana hiburan dalam misinya menawarkan suasana yang kondusif bagi para pengunjung untuk menghabiskan waktunya dengan bersantai.

III. ANALISIS DAN PEMBAHASAN

A. Konsep Held-Karp dalam Pusat Perbelanjaan

Dalam melakukan aktivitas perbelanjaan di *Mall* atau semacamnya terkadang konsumer atau pembeli belum mengetahui barang apa yang hendak konsumer beli atau toko apa yang hendak konsumer kunjungi (Misal: Toko Sepatu, Bioskop, Restoran X, dll). Pada masa pandemi ini pencatatan toko/tempat yang hendak dikunjungi ketika berkunjung ke pusat perbelanjaan akan menjadi solusi yang sangat efektif untuk mengurangi kontak fisik antar konsumer dan meminimalisir waktu kunjung konsumer. Mengapa? Metode ini akan mengarahkan konsumer menuju toko-toko yang hendak dikunjungi serta memberikan rute terpendek juga tercepat. Oleh karena itu, konsumer tidak perlu bingung lagi mencari-cari toko-toko yang konsumer inginkan.

Penerapan metode ini dapat dengan mudah dilakukan dengan pendekatan pada permasalahan *Traveling Salesman Problem*. Permasalahan ini sudah ada sejak tahun 1900-an dan pada pertama kali permasalahan ini muncul, solusi eksak permasalahan ini memiliki kompleksitas yang cukup lama yaitu O(N!) dengan N jumlah titik (dalam kasus ini N adalah jumlah toko), jelas solusi ini sangat tidak layak dipraktikan dalam aplikasi/program nyata yang akan digunakan banyak orang karena sangatlah lambat. Namun, pada tahun 1962, ilmuwan bernama Michael Held dan Richard Karp menciptakan algoritma bernama *Held-Karp Algorithm* yang memiliki kompleksitas jauh lebih cepat daripada solusi eksak awalnya. Kecepatannya memiliki kasus terburuk $O(2^N N^2)$.

Gambar 3.1 *Michael Held, Richard Shareshian, Richard Karp. IBMArchives.*
(Sumber: http://www.math.uwaterloo.ca/tsp/uk/history.html)

Algoritma *Held-Karp* layak untuk dipraktikan pada sebuah program, karena dianggap mampu untuk mencari rute terpendek dengan waktu tercepat. Sebagai perbandingan kecepatan Held-Karp, berikut adalah tabel perbandingan algoritma TSP.

TSP OF 100 CITIES, ALGORITHM COMPARISON

Algorithm	Optimal route length (km)	Elapsed time (sec)	Iterations
Nearest Neighbor	26664	2.5	100
Genetic	25479	45	10000
Greedy Heuristic	23311	0.07	18

TABLE II
TSP OF 1000 CITIES, ALGORITHM COMPARISON

Algorithm	Optimal route length (km)	Elapsed time (sec)	Iterations
Nearest Neighbor	83938	95.5	1000
Genetic	282866	468	10000
Greedy Heuristic	72801	127	151

Tabel 3.1 *Perbandingan Algoritma TSP*
(Sumber: Comparison of Algorithms for Solving Traveling Salesman Problem [6])

Pada tabel diatas, *Greedy Heuristic* merupakan algoritma *Held-Karp*. Pada penggunaan 100 vertex, algoritma *Held-Karp* jauh lebih cepat dibandingkan algoritma *Genetic* dan algoritma *Nearest Neighbor*. Namun, pada penggunaan 1000 vertex, algoritma *Held-Karp* sedikit lebih lambat jika dibandingkan dengan algoritma *Nearest Neighbor*. Lalu mengapa pada kasus pusat perbelanjaan akan lebih efektif jika menggunakan algoritma *Held-Karp* jika dibandingkan dengan algoritma *Nearest Neighbor*. Karena pada rata-rata daftar kunjungan belanja konsumer pada pusat perbelanjaan kurang dari 100. Sehingga algoritma *Held-Karp* dapat lebih cepat dalam mencari rute terpendek.

B. Algoritma dan Penerapan

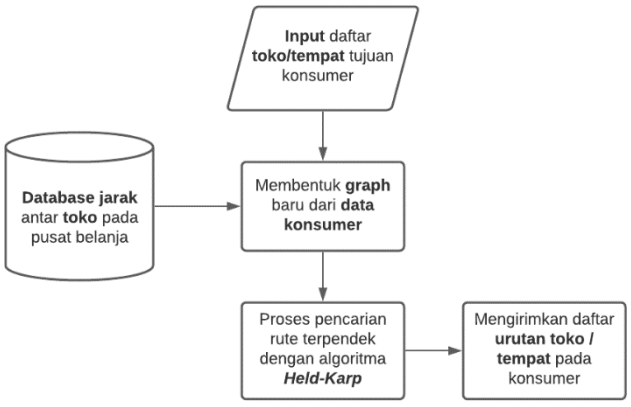
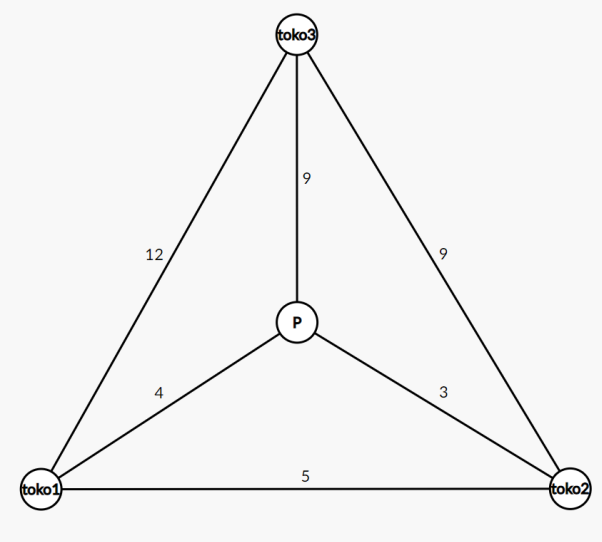


Diagram 3.1 *Design pembentukan daftar urutan toko/tempat yang harus dilalui konsumer*
(Sumber: Dokumen Penulis)

Pada penerapan algoritma *Held-Karp* diperlukan data toko atau tempat yang hendak dikunjungi konsumer terlebih dahulu. Selain itu, setiap pusat perbelanjaan harus memiliki *database* yang berisi jarak antar toko pada pusat perbelanjaan tersebut.

Database ini berguna untuk mengirimkan data kepada graf yang akan dibentuk berdasarkan toko atau tempat yang dikirimkan oleh konsumer. Setiap vertex pada graf ini akan memiliki edge ke setiap vertex lainnya, sehingga jika terdapat N buah toko, maka akan terdapat N+1 vertex (ditambah edge pintu masuk) dan $\frac{(N-1)N}{2}$ sisi atau edge pada graf tersebut. Dengan asumsi konsumer akan memasuki pusat belanja melalui pintu masuk lalu akan kembali lagi untuk pulang melalui pintu tersebut juga dan graf sudah berhasil dibentuk dan memiliki bobot pada setiap edgenya (bobot mengartikan jarak antar toko pada kasus ini) maka graf sudah siap untuk diproses pada algoritma *Held-Karp*. Untuk mempermudah dalam proses mencari rute terpendek, berikut adalah ilustrasi graf dengan 3 toko tujuan konsumer.



Gambar 3.2 *Graf berisi 3 toko dan 1 pintu masuk beserta jaraknya.*
(Sumber: Dokumen Penulis)

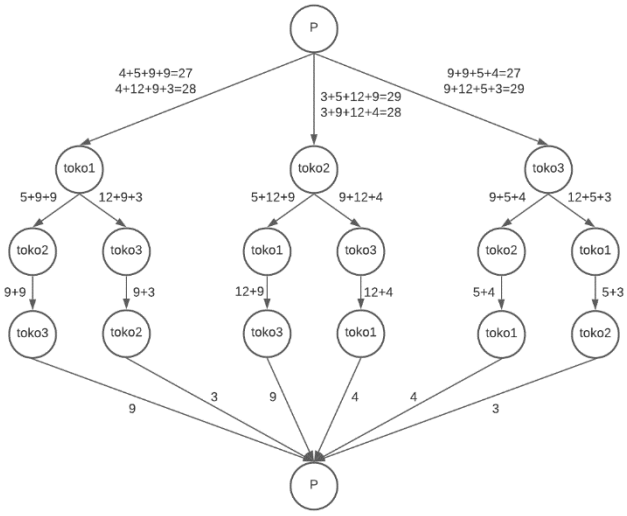
Dalam graf tersebut, P merupakan pintu masuk dan toko1, toko2, dan toko3 merupakan toko-toko tujuan konsumer. Pada kasus pusat perbelanjaan akan diasumsikan jarak dari toko A ke toko B dan toko B ke toko A adalah sama dan konsumer hanya dapat melewati daftar toko yang dia inginkan sebanyak satu kali maksimal. Untuk menjalankan algoritma *Held-Karp*, pertama-tama diperlukan tabel berisikan jarak antar 2 toko sebagai berikut.

Jarak	P	toko1	toko2	toko3
P	0	4	3	9
toko1	4	0	5	12
toko2	3	5	0	9
toko3	9	12	9	0

Tabel 3.2 *Tabel jarak antar toko dan pintu masuk*
(Sumber: Dokumen Penulis)

Perhatikan bahwa tabel tersebut seperti dicerminkan, hal ini diakibatkan karena jarak antar dua toko pada kedua arahnya bernilai sama.

Setelah ini akan dibentuk struktur data tree yang dimulai dari P hingga kembali lagi ke P dengan melewati setiap toko.



Gambar 3.3 *Pohon hasil dari graf toko*
(Sumber: Dokumen Penulis)

Algoritma *Held-Karp* akan mengiterasi setiap jarak dari bawah atau dari langkah terakhir yaitu saat kembali ke pintu masuk (P). Lalu akan menyimpannya datanya seperti ini,

```
g(toko1, {})=4
g(toko2, {})=3
g(toko3, {})=9
g(toko1, {toko2})=21
g(toko1, {toko3})=8
g(toko2, {toko1})=9
g(toko2, {toko3})=18
g(toko3, {toko1})=16
g(toko3, {toko2})=12
g(toko1, {toko2,toko3})= min(23,24)=23
g(toko2, {toko3,toko1})= min(26,25)=25
g(toko3, {toko1,toko2})= min(18,20)=18
g(P,{toko1,toko2,toko3})= min(27,28,27)=27
```

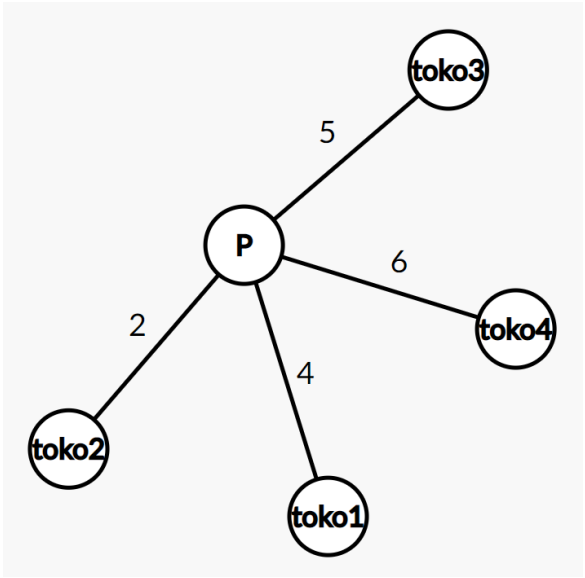
Fungsi g(vertex **start**, vertex **listOfToko[]**) adalah fungsi yang menyimpan jarak terpendek dari *start* dan melewati semua toko yang berada pada *listOfToko* lalu kembali lagi ke *start*. Ketika dibuat algoritma pada program sesungguhnya, fungsi ini akan diubah menjadi sebuah tipe data yang menyimpan nilai, sehingga bisa dilakukan pemrograman dinamis di dalam data tersebut. Rute terpendek ditemukan jika g(P, {seluruh toko}) sudah didapatkan hasilnya dan hasil rute terpendeknya adalah nilai dari g(P, {seluruh toko}) sendiri.

Setelah dilakukan pemrosesan pada algoritma *Held-Karp*, didapatkan hasil jarak terpendeknya yaitu 27 dengan menjalankan langkah P-toko1-toko2-toko3-P atau sebaliknya P-toko3-toko2-toko1-P.

Algoritma *Held-Karp* ini menggunakan teknik pemrograman dinamis untuk menyimpan jarak-jarak terpendek di sub-graf nya sebelum pada akhirnya mendapatkan hasil jarak terpendek keseluruhan. Teknik sangat efektif jika terdapat banyak pengulangan perhitungan pada suatu fungsi. Dengan adanya pemrograman dinamis, kompleksitas akan dipangkas sehingga menjadi sangat cepat dibandingkan dengan rekursif biasa.

IV. Beberapa Kesalahan Umum

Misalkan konsumer hendak menuju ke 4 toko berbeda dan saat ini konsumer berada di pintu masuk.



Gambar 4.1 *Vertex P dengan semua edgenya*
(Sumber: Dokumen Penulis)

Kebanyakan kesalahan umum pada algoritma adalah dengan mencari langsung toko yang memiliki jarak terpendek lalu menghapus toko tersebut dari daftarnya, lalu mencari lagi toko lainnya dengan jarak terpendek, dst. Teknik berikut tidak menjamin bahwa konsumer akan melewati rute terpendeknya, karena mungkin saja langkah pertamanya bukan toko terdekat namun total jarak yang ditempuh pada konsumer pada akhir kembali ke pintu masuk lebih pendek.

V. Kesimpulan

Penerapan algoritma *Held-Karp* ini akan sangat efektif jika diterapkan pada seluruh pusat belanja di Indonesia. Konsumer hanya perlu membuka aplikasi pada gawai mereka lalu menuliskan daftar toko yang hendak mereka kunjungi dan aplikasi tersebut akan mengeluarkan hasil rute terpendek yang konsumer perlu kunjungi satu per satu. Dengan adanya fitur ini pada aplikasi pemerintah, maka pemerintah dapat lebih gencar dalam melakukan pencegahan dan penularan virus selama pandemi di masyarakat.

VI. Ucapan Terima Kasih

Penulis mengucapkan syukur dan terima kasih kepada Tuhan Yang Maha Esa beserta seluruh Bapak dan Ibu dosen pengampu mata kuliah IF 2120 Matematika Diskrit karena atas berkat dan rahmatnya, Penulis dapat menyelesaikan makalah berjudul "Penerapan Algoritma Held-Karp di Pusat Belanja dalam Meminimalisir Waktu dan Kontak Fisik Konsumer selama Pandemi". Penulis juga mengucapkan terima kasih kepada teman-teman yang telah mendukung penulis selama proses pengerjaan makalah ini.

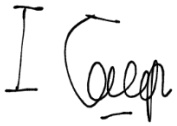
Daftar Pustaka

[1] Munir, Rinaldi. 2021. Slide Kuliah IF2120 Matematika Diskrit Graf, Bandung.
[2] Amelia, Windy. 2012. Implementasi Teori Logika dan Graf dalam Menentukan Efisiensi Rangkaian Listrik, Bandung.
[3] Tang, Daisy. 2012. CS241 – Lecture Notes: Graphs, Pomona.
[4] Risfanani, Muhammad Angga. 2019. Penggunaan *Graph* Dalam Pengefisienan Proses Penyaluran Barang Ekspedisi Kargo/Paket, Bandung.
[5] Wengrow, Jay. 2020. A Common-Sense Guide to Data Structures and Algorithms, Second Edition.
[6] Abdulkarim, Haider A. & Alshammari, Ibrahim F. 2015. Comparison of Algorithms for Solving Traveling Salesman Problem, *International Journal of Engineering and Advanced Technology (IJEAT)*.
[7] Chiara, J. D. & Crosbie , M. J., 2001. Time Saver Standart For Building Types. 4th penyunt. Singapore: McGraw - Hill Book Co.

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



I Gede Arya Raditya Parameswara
13520036