Aplikasi *Travelling Salesman Problem* untuk Perencanaan Rute Pengiriman Produk Mister Donut ke Cabang Indomaret di Kota Bandung

Rachel Gabriela Chen - 13521044
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13521044@std.stei.itb.ac.id

***Abstract*—Mister Donut adalah sebuah *franchise* donat asal Jepang yang menyuplai donat ke gerai Indomaret, sebuah jaringan minimarket yang beroperasi di Indonesia. Di Bandung, terdapat ± 300 gerai Indomaret yang menyediakan produk Mister Donut. Pengiriman produk dari rumah produksi Mister Donut dilakukan dengan truk. Dalam sekali pengiriman, setiap truk mengirim produk ke beberapa cabang Indomaret yang berada dalam satu daerah. Untuk memaksimalkan keuntungan, diperlukan optimisasi pengiriman produk. Makalah ini membahas perencanaan rute pengiriman produk Mister Donut ke cabang-cabang Indomaret di Bandung dengan memilih rute terpendek melalui pendekatan *Travelling Salesman Problem* (TSP) yang diselesaikan dengan *Dynamic Programming*.**

***Keywords*—Graf, Indomaret, Rute Terpendek, *Travelling Salesman Problem*.**

I. PENDAHULUAN

Indomaret adalah salah satu jaringan minimarket terbesar di Indonesia yang menyediakan kebutuhan pokok dan rumah tangga. Terdapat sekitar 800 cabang Indomaret yang tersebar di berbagai daerah di Bandung Raya. Indomaret menyediakan beberapa produk eksklusif yang menjadi khas Indomaret. Salah satunya adalah donat Mister Donut, sebuah *franchise* donat yang berasal dari Jepang yang tersedia di sejumlah gerai Indomaret di seluruh Indonesia.

Di Bandung Raya, terdapat 204 gerai Indomaret yang menyediakan donat dari Mister Donut. Pengiriman produk Mister Donut ke cabang-cabang Indomaret dilakukan setiap hari mempertimbangkan masa kadaluarsa donat yang singkat (1-2 hari). Pengiriman ini dilakukan dengan menggunakan truk yang telah ditugaskan untuk sejumlah cabang Indomaret yang berada di daerah yang berdekatan. Setiap truk berangkat dari rumah produksi Mister Donut yang berada di kecamatan Dago menuju 10-15 cabang Indomaret dalam sekali pengiriman. Dalam proses pengiriman, terdapat banyak rute yang dapat dipilih agar truk dapat mengirim ke semua cabang yang telah ditentukan kemudian kembali ke rumah produksi.

Untuk meminimalkan biaya distribusi, diperlukan optimisasi pengiriman yang dapat dilakukan dengan menentukan rute terpendek yang dapat dipilih oleh kurir truk saat melakukan pengiriman. Rute terpendek dapat meminimalkan jarak yang ditempuh dan waktu pengiriman. Rumah produksi Mister Donut dan setiap cabang Indomaret yang akan dikunjungi dimodelkan sebagai simpul dalam sebuah graf berbobot. Setiap simpul dihubungkan dengan simpul lain dengan sebuah sisi yang menyatakan jarak antara simpul tersebut. Dengan pemodelan ini, permasalahan dapat diselesaikan dengan menggunakan pendekatan *Travelling Salesman Problem* (TSP).

II. LANDASAN TEORI

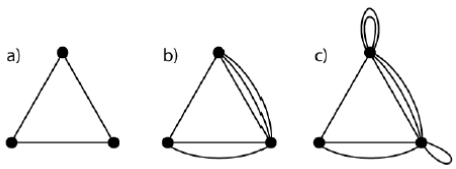
A. Graf

Graf didefinisikan sebagai struktur diskrit berupa kumpulan simpul (*vertices*) yang terhubung melalui himpunan sisi (*edges*). Graf disajikan dalam bentuk $G = (V, E)$, dimana G adalah graf, V adalah himpunan tidak kosong simpul-simpul $v_1, v_2, \ldots, v_n$, dan E adalah himpunan sisi $e_1, e_2, \ldots, e_n$, yang menghubungan sepasang simpul dalam graf.

Sepasang simpul dalam graf dapat dihubungkan dengan dua sisi yang berbeda, pasangan sisi ini disebut sisi ganda (*multiple edges*). Terdapat juga sisi yang berawal dan berakhir pada simpul yang sama yang dinamakan gelang atau kalang (*loop*).

Berdasarkan keberadaan sisi ganda atau sisi gelang, graf dapat dikelompokkan menjadi:

1. Graf sederhana (simple graph), yaitu graf yang tidak memiliki sisi ganda dan sisi gelang.
2. Graf tak-sederhana (unsimple graph), yaitu graf yang mengandung sisi ganda atau gelang. Graf tak-sederhana yang memiliki gelang disebut graf-semu.



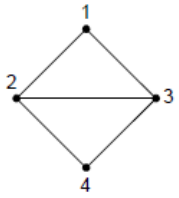
Gambar 2.1 (a) Graf sederhana (b) Graf tak-sederhana (c) Graf-semu (Sumber : [1])

Berdasarkan orientasi arah sisi, graf dapat dikelompokkan menjadi:

1. Graf tak berarah (undirected graph), yaitu graf yang sisinya tidak mempunyai orientasi arah.
2. Grah berarah (directed graph), yaitu graf yang setiap sisinya diberikan orientasi arah.

Beberapa terminologi dalam teori graf:

1. Ketetanggan (adjacency)
   Dua buah simpul dalam sebuah graf bertetangga bila kedua simpul tersebut terhubung oleh setidaknya satu sisi.

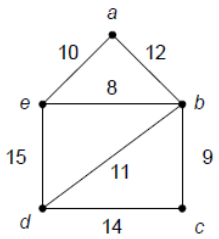


G_1

Gambar 2.2 Contoh graf G1 (Sumber: [1])

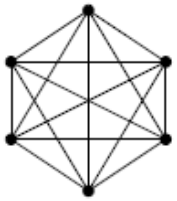
   Pada graf G1 di gambar 2.2, simpul 1 dan simpul 2 bertetangga, begitu juga simpul 2 dan simpul 3.

2. Bersisian (incidency)
   Sebuah sisi *e* dikatakan bersisian dengan simpul v_i dan v_j jika sisi tersebut menghubungkan simpul v_i dengan v_j.
3. Derajat (degree)
   Derajat suatu simpul adalah jumlah sisi yang bersisian dengan simpul tersebut.
4. Lintasan (path)
   Lintasan dengan panjang ialah barisan simpul-simpul dan sisi-sisi yang dilaluinya v_0,e_1,v_1,e_2,v_2,…,e_n, v_n yang menghubungkan simpul awal v_0 ke simpul tujuan v_n. Panjang lintasan *n* adalah jumlah sisi dalam lintasan tersebut.
5. Siklus (cycle atau circuit)
   Siklus atau sirkuit adalah lintasan yang berawal dan berakhir pada simpul yang sama.
   Pada graf G1 di gambar 2.2, terdapat sirkuit 1 – 2 – 4 – 1 dengan panjang 3.
6. Keterhubungan (connected)
   Dua simpul v_0 dan v_1 dikatakan terhubung jika terdapat lintasan yang menghubungkan v_0 ke v_1.
7. Graf berbobot (weighted graph)
   Graf berbobot adalah graf yang setiap sisinya memiliki nilai atau bobot spesifik.



Gambar 2.3 Contoh graf berbobot (Sumber: [3])

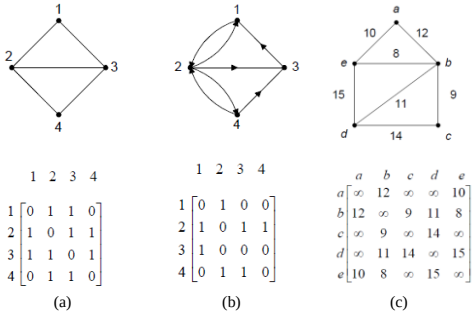
8. Graf lengkap (completed graph)
   Graf lengkap adalah sebuah grah yang memiliki sisi yang menghubungkan setiap simpul dengan semua simpul lainnya dalam graf tersebut.



Gambar 2.4 Contoh graf lengkap dengan jumlah simpul 5 (Sumber: [1])

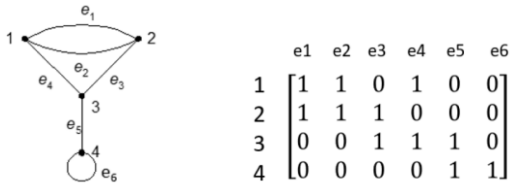
Beberapa cara untuk merepresentasikan graf dengan matriks:

1. Matriks ketetanggaan (adjacency matriks)
   Sebuah graf tidak berarah dengan *n* buah simpul direpresentasikan dengan matriks ketetanggaan M *n x n* dimana elemen pada baris ke-*i* kolom ke-*j* bernilai 1 jika ada sisi yang menghubungkan simpul ke-*i* dengan simpul ke-*j*, 0 jika tidak. Untuk graf berarah, elemen pada baris ke-*i* kolom ke-*j* bernilai 1 jika ada sisi dari simpul ke-*i* ke simpul ke-*j*, 0 jika tidak. Untuk graf berbobot, elemen pada baris ke-*i* kolom ke-*j* adalah bobot sisi yang menghubungkan simpul ke-*i* dengan simpul ke-*j*.



Gambar 2.5 Representasi matriks ketetanggaan a)graf tidak berarah, b) graf berarah c) graf berbobot (Sumber: [2])

2. Matriks bersisian (incidency matrix)
   Sebuah graf tidak berarah dengan *n* buah simpul dan *m* buah sisi direpresentasikan dengan matriks ketetanggaan M *n x m* dimana elemen pada baris ke-*i* kolom ke-*j* bernilai 1 jika simpul ke-*i* bersisian dengan sisike-*j*, 0 jika tidak.



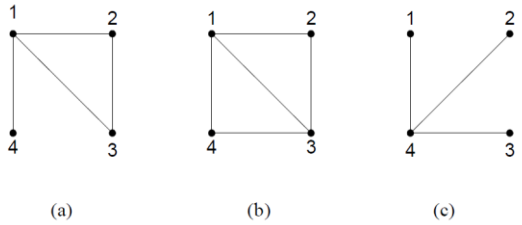
Gambar 2.6 Representasi matriks bersisian graf (Sumber: [2])

B. Lintasan dan Sirkuit Hamilton

Lintasan Hamilton adalah istilah yang digunakan dalam teori graf untuk sebuah lintasan yang melalui setiap simpul dalam graf sebanyak satu kali saja. Lintasan ini dinamai menurut matematikawan Sir William Rowan Hamilton yang pertama kali mengemukakan teori mengenai lintasan ini. Lintasan

Hamilton digunakan untuk menemukan jalur terpendek antara dua titik dalam suatu sistem.

Sirkuit Hamilton adalah lintasan Hamilton yang melalui setiap titik dalam sistem dan kembali ke titik awal, sehingga membentuk sebuah sirkuit. Dengan kata lain, sirkuit Hamilton adalah lintasan yang melalui setiap simpul dalam sebuah graf tepat satu kali, kecuali simpul awalnya yang dilalui dua kali. Graf yang mempunyai sirkuit Hamilton disebut graf Hamilton, sedangkan graf yang mempunyai lintasan Hamilton saja disebut graf semi-Hamilton.



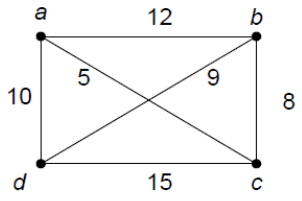
Gambar 2.7 (a) Graf semi-Hamilton (b) Graf Hamilton (c) Bukan Graf semi-Hamilton ataupun graf Hamilton (Sumber: [3])

Sebuah graf sederhana G dengan n (≥ 3) buah simpul adalah graf Hamilton bila: $r \geq n/2$ untuk semua simpul di G dengan r adalah derajat simpul.

Setiap graf lengkap adalah graf Hamilton. Sebuah graf lengkap dengan n buah simpul memiliki total $\frac{(n-1)!}{2}$ buah sirkuit Hamilton. Jika n≥4 dan n genap, maka terdapat $\frac{n-2}{2}$ buah sirkuit Hamilton yang saling lepas. Jika n≥3 dan n ganjil, maka terdapat $\frac{n-1}{2}$ buah sirkuit Hamilton yang saling lepas.

C. Travelling Salesman Problem

Travelling Salesman Problem (TSP) adalah contoh masalah optimisasi yang dapat diselesaikan dengan lintasan Hamilton. Misalnya, diberikan sejumlah kota dan diketahui jarak antarkota. Seorang pedagang harus mengunjungi semua kota dan kembali ke kota asalnya. Dicari jalur terpendek yang dapat dilalui pedagang tersebut untuk menghemat waktu dan biaya perjalanan. Untuk penyelesaian masalah ini, dibuat sebuah graf dengan setiap kota sebagai simpul dan jalan yang menghubungkan tiap kota sebagai sisi. Bobot setiap sisi merupakan jarak antarkota yang dihubungkan oleh sisi tersebut.



Gambar 2.8 Contoh graf berbobot untuk permasalahan TSP

Untuk sebuah graf lengkap, diketahui jumlah sirkuit Hamiltonnya adalah $\frac{(n-1)!}{2}$, sehingga permasalahan ini dapat diselesaikan dengan menghitung bobot setiap sirkuit Hamilton dan memilih bobot minimum. Graf pada gambar 2.6 memiliki $\frac{(4-1)!}{2} = 3$ buah sirkuit Hamilton. Dengan mengecek bobot semua sirkuit, didapatkan rute terpendek a – c – b – d – a dengan jumlah bobot 32. Algoritma ini merupakan algoritma brute-force dengan kompleksitas waktu O(*n!*), dimana *n* adalah jumlah simpul dalam graf lengkap. Untuk *n* < 5, pendekatan ini dapat menyelesaikan masalah dengan mudah. Namun, untuk *n* yang besar, pendekatan ini tidak efisien karena banyaknya kemungkinan sirkuit Hamilton yang perlu dihitung bobotnya. Oleh karena itu, dalam makalah ini, digunakan pendekatan pemrograman dinamis (*dynamic programming*) untuk penyelesaian persoalan TSP untuk mencapai solusi yang efisien.

D. Algoritma Dynamic Programming TSP

Dynamic programming (DP) atau pemrograman dinamis adalah sebuah teknik dalam pemrograman yang digunakan untuk menyelesaikan masalah yang memiliki banyak sub-masalah yang saling terkait. Penyelesaian masalah dengan pemrograman dinamis dilakukan dengan memecah sebuah masalah besar menjadi sub-masalah yang lebih kecil. Solusi dari setiap sub-masalah yang lebih kecil disimpan. Dengan demikian, pemrograman dinamis menghindari perhitungan ulang sub-masalah yang telah dihitung dan mencapai solusi yang lebih efisien. Optimisasi dengan pemrograman dinamis mereduksi kompleksitas waktu dari eksponensial menjadi polinomial.

Pemrograman dinamis dapat digunakan untuk penyelesaian masalah travelling salesman karena dalam $\frac{(n-1)!}{2}$ permutasi kemungkinan rute, terdapat sub-rute yang tumpang tindih (*overlapping*), misalnya rute *A – B – C – (n – 3)* memiliki sub-rute yang sama dengan *A – C – B – (n – 3)* dimana *n – 3* adalah kota-kota lain selain *A – B – C*. Perhitungan ulang sub-rute yang sama dapat dihindari dengan pemrograman dinamis sehingga waktu komputasi dapat dikurangi.

Dengan pendekatan pemrograman dinamis, digunakan sebuah fungsi rekursif *dp(pos, mask)* dengan 2 parameter yaitu *pos*, kota (simpul) terakhir yang dikunjungi dan *mask*, himpunan kota yang telah dikunjungi. Himpunan ini direpresentasikan dengan *bitmask* untuk efisiensi. Misalnya, terdapat *n* kota, maka digunakan sebuah bilangan biner *b* dengan panjang bit *n* untuk merepresentasikan kota-kota tersebut. Jika bit ke-*i* pada *b* bernilai 1, maka kota *i* telah dikunjungi. Sebaliknya jika bit ke-*i* bernilai 2, maka kota *i* belum dikunjungi. Misalnya, diberikan *bitmask* = 18 = 10010_2. Karena bit ke-1 dan bit ke-4 bernilai 1, himpunan kota yang dikunjungi = {1,4}. Fungsi *dp(pos, mask)* mengembalikan total jarak tempuh minimum.

Misalnya terdapat matriks ketetanggan *dist* dimana dist[i][j] adalah bobot sisi yang menghubungkan kota *i* dengan kota *j*. Secara matematis, fungsi dp(pos, mask) dirumuskan:

1. dp(pos, 2n-1) = dist[pos][0]
   Basis dari fungsi rekursif *dp*, yaitu saat semua kota telah dikunjungi. Nilai yang dikembalikan adalah jarak antara kota terakhir yang dikunjungi dengan kota awal.
2. dp(pos, mask) = min(dist[pos][nxt] + dp(nxt, mask | (1 << nxt)))
   Menambahkan jarak kota terakhir dengan kota selanjutnya dengan jarak minimum antarkota selanjutnya dengan sisa kota yang belum dikunjungi.

Berikut adalah conoth *pseudocode* untuk penyelesaian TSP dengan pemrograman dinamis (tanpa implementasi bitmask):

Algorithm 1: Dynamic Programming algorithm for the original TSP

Data: A set of locations V, an arbitrary location $v \in V$ and cost function c

Result: A shortest tour that visits all locations in V

Initialize D_{TSP} with values ∞ ;
Initialize a table P to retain predecessor locations ;
Initialize v as an arbitrary location in V ;
foreach $w \in V$ **do**
    $D_{\text{TSP}}(\{w\}, w) \leftarrow c(v, w)$;
    $P(\{w\}, w) \leftarrow v$;
for $i = 2, \ldots, |V|$ **do**
    for $S \subseteq V$ *where* $|S| = i$ **do**
        foreach $w \in S$ **do**
            foreach $u \in S$ **do**
                $z \leftarrow D_{\text{TSP}}(S \setminus \{w\}, u) + c(u, w)$;
                if $z < D_{\text{TSP}}(S, w)$ **then**
                    $D_{\text{TSP}}(S, w) \leftarrow z$;
                    $P(S, w) \leftarrow u$;
return *path obtained by backtracking over locations in* P *starting at* $P(V, v)$;

Gambar 2.9 Pseudocode Penyelesaian TSP dengan Pemrograman Dinamis (Sumber: [4])

III. METODOLOGI

A. Batasan Masalah

Dalam makalah ini, terdapat beberapa batasan yang digunakan untuk menyederhakan masalah penentuan rute pengiriman produk Mister Donut ke Cabang Indomaret di Bandung. Batasan-batasan tersebut antara lain:

1. Rute optimum hanya ditentukan oleh jarak tempuh minimum. Faktor-faktor seperti kemacetan, kualitas jalan, waktu pengiriman dilakukan, dan waktu tempuh diabaikan.
2. Data jarak antarlokasi yang digunakan merupakan jarak minimum.
3. Setiap truk telah ditugaskan untuk melakukan pengiriman ke *n* buah cabang Indomaret yang terletak di daerah yang berdekatan sehingga tidak perlu dilakukan pemilihan cabang-cabang Indomaret untuk masing-masing truk.
4. Setiap truk mengirim ke maksimal 15 cabang Indomaret.

B. Data yang Digunakan

Data yang digunakan berasal dari arsip data Indomaret Bandung Raya yang didapatkan dari observasi lapangan. Data ini memuat 204 cabang Indomaret di Bandung Raya yang menerima pengiriman donat dari Mister Donat. Informasi dari setiap cabang mencakup kode toko, nama toko, domain, subdomain, titik koordinat, alamat, kota, dan kode pos.

	KD	NAMA TOKO	DOMAIN	SUBDOMAIN	TITIK KOORDINAT	ALAMAT	KELURAHAN	KECAMATAN	KOTA	KODE POS
1	TOYJ	FRESH HOTEL PREANGER	TRAFFIC	HOTEL/PENGINAPAN	S6 55 17.06 E107 36 42.26	JL. ASIA AFRIKA NO. 81	BRAGA	SUMUR BANDUNG	KOD. BANDUNG	40111.0
2	TKUT	POINT BRAGA	TRAFFIC	RESIDENTIAL	S6 55 03.8 E107 36 34.2	JL. BRAGA NO. 52 BRAGA SUMUR BANDUNG KOTA BA...	BRAGA	SUMUR BANDUNG	KOD. BANDUNG	40111.0
3	T4C3	SUNIARAJA 25	TRAFFIC	RESIDENTIAL	S6 54 58.8 E107 36 28.6	JL. SUNIARAJA NO. 25 D RT 01 RW 04 BRAGA SUMU...	BRAGA	SUMUR BANDUNG	KOD. BANDUNG	40111.0
4	TX0M	FRESH SUNDA 89	TRAFFIC	RESIDENTIAL	S6 55 1.52 E107 37 4.47	JL. SUNDA NO.89 KEBON PISANG SUMUR BANDUNG KOT...	KEBON PISANG	SUMUR BANDUNG	KOD. BANDUNG	40112.0
5	TAO6	FRESH JL JAWA	RESIDENTIAL	TRAFFIC	S6 54 56.15 E107 36 56.71	JL. JAWA NO. 40	MERDEKA	SUMUR BANDUNG	KOD. BANDUNG	40113.0
...	...	...	...	...	...	...	...	...	...	...

Gambar 3.1 Cuplikan data cabang Indomaret di Bandung yang menyediakan produk Mister Donut (Sumber : Observasi Lapangan dan Dokumen Informan)

Lokasi awal pengiriman (rumah produksi Mister Donut) berada di Mister Donut Dago Jl. Ir. H. Juanda No. 44, Dago Bawah, Bandung.

C. Pemodelan Masalah

Langkah 1

Dipilih *n* buah cabang Indomaret yang berada pada lokasi yang berdekatan untuk ditugaskan kepada sebuah truk. Dalam makalah ini, hanya akan ditunjukkan penentuan rute optimum untuk 1 buah truk.

Berikut adalah data cabang-cabang Indomaret yang dipilih untuk 1 *shift* pengiriman:

Kode Toko	Nama Toko	Alamat
TOYJ	Fresh Hotel Preanger	Jl. Asia Afrika No. 81
TKUT	Point Braga	Jl. Braga No. 52 Braga Sumur Bandung Kota Bandung
T4C3	Suniaraja 25	Jl. Suniaraja No. 25 D Rt 01 Rw 04 Braga Sumur Bandung Kota Bandung
TX0M	Fresh Sunda 89	Jl. Sunda No.89 Kebon Pisang Sumur Bandung Kota. Bandung
TAO6	Fresh Jl Jawa	Jl. Jawa No. 40
TST4	Ahmad Yani 233	Jl.Ahmad Yani No.233 Merdeka Sumur Bandung Kota Bandung
TL2Z	Point Jl. Ambon	Jl. Ambon
FNLZ	Bengawan	Jl. Bengawan Cihapit Bandung Wetan Kota Bandung
TB79	Ahmad Yani 269	Jl. Jend. A. Yani No. 269 Rt 01 Rw 08 Cihapit Bandung Wetan Kota Bandung
T0BX	Cihampelas 70	Jl.Cihampelas No. 70 Rt 02 Rw 09 Tamansari Bandung Wetan Kota Bandung
F308	YPAC	Jl. Taman Sari No. 31 Rt 07 Rw 12 Tamansari Bandung Wetan Kota Bandung
TT11	Cicendo	Jl. Cicendo No. 6 A Rt. 04 Rw. 02 Babakan Ciamis Sumur Bandung Kota Bandung
T3LU	Aceh	Jl. Aceh No. 41 Rt 01 Rw 04 Babakan Ciamis Sumur Bandung Kota Bandung
T9UN	Ahmad Yani 363	Jl. Jend A. Yani No. 363-365 Sukamaju Cibeunying Kidul Kota Bandung

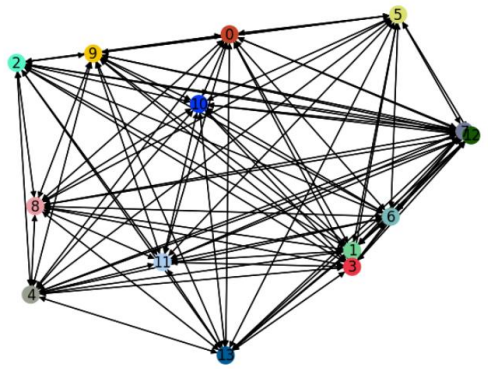
Tabel 3.1 Cabang Indomaret yang berada dalam shift pengiriman yang sama

Langkah 2

Dari data lokasi cabang Indomaret yang perlu menerima pengiriman, dicari jarak antara cabang yang satu dengan semua cabang yang lain. Dicari juga jarak antara setiap cabang dengan rumah produksi Mister Donut. Pencarian jarak dilakukan menggunakan Google Maps Distance Matrix API sehingga data yang didapatkan merupakan jarak terpendek antara dua lokasi yang dapat dilewati oleh truk.

Data yang didapatkan kemudian dimodelkan sebagai graf berbobot dengan masing-masing lokasi sebagai simpul dan setiap sisi memiliki bobot sesuai jarak antarlokasi. Berikut

adalah pemodelan permasalahan sebagai graf berbobot (bobot tidak ditampilkan karena tumpang tindih) :



Gambar 3.2 Pemodelan permasalahan sebagai graf (Sumber: Dokumen Pribadi)

Perhatikan bahwa graf di atas adalah graf berarah dan berbobot karena jarak yang ditempuh untuk berangkat dari lokasi *i* ke lokasi *j* berbeda dengan jarak dari lokasi *j* ke lokasi *i*.

Nomor Simpul	Kode Toko	Nomor Simpul	Kode Toko
0	Mister Donut	7	TST4
1	TOYJ	8	TL2Z
2	TKUT	9	FNLZ
3	T4C3	10	TB79
4	TX0M	11	T0BX
5	TA06	12	F308
6	TST4	3	TT11

Tabel 3.2 Pemetaan simpul-simpul dalam graf ke cabang Indomaret

Representasi graf dalam bentuk matriks ketetanggaan dimana setiap sel mewakili jarak (dalam meter):

DARI/KE	MISTER DONUT	TOYJ	TKUT	T4C3	TX0M
MISTER DONUT	∞	2494	2213	1898	2521
TOYJ	3674	∞	823	1085	1914
TKUT	2852	1079	∞	262	1611
T4C3	2590	1612	1331	∞	2144
TX0M	2500	1528	1769	2399	∞
TA06	2666	1084	1325	1955	863
TST4	3484	2638	2879	3569	2416
TL2Z	1487	2084	2325	2101	1863
FNLZ	2561	3030	3544	3565	2450
TB79	3845	2534	3048	3310	1954
T0BX	610	3098	2817	2502	3125
F308	2166	3008	2727	2412	3035
TT11	3444	2466	2185	1658	2998
T3LU	2599	1574	1815	2445	1353

DARI/KE	TA06	TST4	TL2Z	FNLZ	TB79
MISTER DONUT	1938	3465	1409	2971	3578
TOYJ	2358	2606	3004	3723	3384
TKUT	2055	2303	2181	3302	3081
T4C3	1956	2835	1919	3040	3613
TX0M	444	1351	1267	1936	1879
TA06	∞	1513	1222	2056	2042
TST4	1554	0	2107	1374	778
TL2Z	1171	2067	∞	1517	2307
FNLZ	2121	1219	1672	∞	876
TB79	2049	1147	2342	924	∞
T0BX	2542	3670	1713	3139	4015
F308	2452	3979	1942	3537	4327
TT11	2810	3690	2774	3895	4467
T3LU	490	792	1222	1908	1569

DARI/KE	T0BX	F308	TT11	T3LU
MISTER DONUT	684	1557	1605	2428
TOYJ	3446	4009	2793	2241
TKUT	2623	3187	1970	1938
T4C3	2361	2925	1708	2446
TX0M	2641	3932	3375	327
TA06	2807	3887	2931	490
TST4	3722	4825	4544	1325
TL2Z	1594	2830	3077	1222
FNLZ	2745	3643	4001	1893
TB79	4029	4927	5040	1821
T0BX	∞	891	1652	2825
F308	1483	∞	1547	2942
TT11	3215	3220	∞	3300
T3LU	2687	3943	3421	∞

Tabel 3.3 Representasi graf permasalahan dengan matriks ketetanggaan graf berbobot

D. Implementasi Dynamic Programming untuk TSP dalam Bahasa Python

Berdasarkan landasan teori pada bab sebelumnya, penulis membuat implementasi algoritma penyelesaian TSP dengan pendekatan pemrograman dinamis menggunakan bahasa pemrograman Python. Implementasi ini dimodifikasi agar fungsi *dp(pos, bitmask)* tidak hanya menyimpan total jarak minimum, tetapi juga rute yang ditempuh untuk mendapatkan jarak minimum tersebut.

Inisialisasi variabel-variabel yang diperlukan. *n* yaitu jumlah cabang Indomaret (simpul) yang perlu dikunjungi, memo dan parent yaitu *list* berukuran *n+1* yang diinisialisasi dengan nilai -1. *List memo* digunakan untuk menyimpan jarak terpendek untuk masing-masing sub-rute. *List parent* digunakan untuk menyimpan simpul-simpul yang dilewati untuk mendapatkan sub-rute terpendek. Inisialisai juga graf yang direpresentasikan sebagai matriks ketetanggaan.

```
n = 13
memo = [[-1]*(1 << (n+1)) for _ in range(n+1)]
parent = [[-1]*(1 << (n+1)) for _ in range(n+1)]
graph = [
        [0, 2494, 2213, 1898, 2521, 1938, 3465, 1409, 2971, 3578, 684, 1557, 1605, 2428],
        [3674, 0, 823, 1085, 1914, 2358, 2606, 3004, 3723, 3384, 3446, 4009, 2793, 2241],
        [2852, 1079, 0, 262, 1611, 2055, 2303, 2181, 3302, 3081, 2623, 3187, 1970, 1938],
        [2590, 1612, 1331, 0, 2144, 1956, 2835, 1919, 3040, 3613, 2361, 2925, 1708, 2446],
        [2500, 1528, 1769, 2399, 0, 444, 1351, 1267, 1936, 1879, 2641, 3932, 3375, 327],
        [2666, 1084, 1325, 1955, 863, 0, 1513, 1222, 2056, 2042, 2807, 3887, 2931, 490],
        [3484, 2638, 2879, 3569, 2416, 1554, 0, 2107, 1374, 778, 3722, 4825, 4544, 1325],
        [1487, 2084, 2325, 2101, 1863, 1171, 2067, 0, 1517, 2307, 1594, 2830, 3077, 1222],
        [2561, 3030, 3544, 3565, 2450, 2121, 1219, 1672, 0, 876, 2745, 3643, 4001, 1893],
        [3845, 2534, 3048, 3310, 1954, 2049, 1147, 2342, 924, 0, 4029, 4927, 5040, 1821],
        [610, 3098, 2817, 2502, 3125, 2542, 3670, 1713, 3139, 4015, 0, 891, 1652, 2825],
        [2166, 3008, 2727, 2412, 3035, 2452, 3979, 1942, 3537, 4327, 1483, 0, 1547, 2942],
        [3444, 2466, 2185, 1658, 2998, 2810, 3690, 2774, 3895, 4467, 3215, 3220, 0, 3300],
        [2599, 1574, 1815, 2445, 1353, 490, 792, 1222, 1908, 1569, 2687, 3943, 3421, 0]
        ]
```

Penyelesaian permasalahan dilakukan dengan implementasi fungsi dp(*pos, bitmask*) dimana *pos* adalah kota terakhir yang dikunjungi dan *bitmask* adalah bilangan biner yang menyimpan kota-kota yang telah dikunjungi.

```python
def dp(pos, bitmask):
    """ Basis """
    if(bitmask == (1<<n+1) - 1):
        return graph[pos][0]
    """ Jika sub-rute telah dihitung sebelumnya, kembalikan hasil yang telah dimemoisasi """
    if (memo[pos][bitmask] != -1):
        return memo[pos][bitmask]
    ans = 10**9
    """ Rekurens """
    for nxt in range (n+1):
        if nxt!=pos and not (bitmask & (1<<nxt)):
            cur_value = graph[pos][nxt] + dp(nxt, bitmask | (1 << nxt))
            if (ans>cur_value):
                ans = cur_value
                parent[pos][bitmask] = nxt
    """ Simpan solusi sub-rute dalam memo """
    memo[pos][bitmask] = ans
    return ans

```

Simpul awal dimulai dengan simpul 0 sehingga hanya terdapat 1 simpul yang telah dikunjungi. Maka, *pos* = 0 dan *bitmask* = 00000000000001_2 = 1. Jarak minimum dihitung dengan dp(0,1).

```python
""" Mulai dengan simpul ke-0, dalam hal ini Mister Donut Dago """
distance = dp(0,1)
```

Cari sirkuit Hamilton dengan rute terpendek dengan menggunakan nilai yang telah dicatat dalam *list parent* kemudian cetak hasil.

```python
""" Cari path minimum """
path = [0 for _ in range (n+1)]
path_ctr = 0
cur_node = 0
cur_mask = 1
while (cur_node!=-1):
    path[path_ctr] = cur_node
    path_ctr +=1
    cur_node = parent[cur_node][cur_mask]
    if (cur_node != -1):
        cur_mask = cur_mask | (1 << cur_node)
path.append(0)

print("Rute paling optimum: ", path)
print("Jarak yang ditempuh: ", distance, "m")
```

IV. Analisis dan Pembahasan

A. Hasil Eksekusi Program Penyelesaian TSP dengan Pemrograman Dinamis

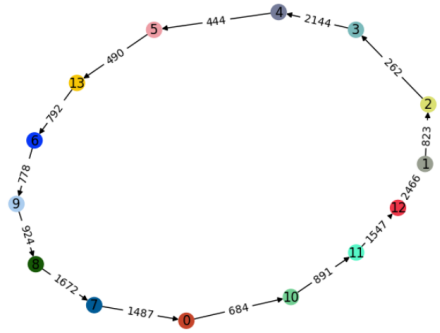
Hasil eksekusi program penyelesaian TSP dengan pemrograman dinamis untuk merencanakan rute optimum pengiriman produk Mister Donut di Kota Bandung:

```
Rute paling optimum:  [0, 10, 11, 12, 1, 2, 3, 4, 5, 13, 6, 9, 8, 7, 0]
Jarak yang ditempuh:  15404 m
Execution time: 0.26502561569213867 seconds
```

Rute optimum pengiriman produk Mister Donut dari rumah produksi ke cabang-cabang Indomaret yang dipilih:

0. Rumah Produksi Mister Donut – Mister Donut Dago
1. T0BX - Jl.Cihampelas No. 70 RT 02 RW 09. Tamansari
2. F308 - Jl. Taman Sari No. 31 RT 07 RW 12. Tamansari
3. TT11 - Jl. Cicendo No. 6 A RT 04 RW 02
4. TOYJ - Jl. Asia Afrika No. 81
5. TKUT - Jl. Braga No. 52, Braga
6. T4C3 - Jl. Suniaraja No. 25 D RT 01 RW 04, Braga
7. TX0M - Jl. Sunda No.89, Kebon Pisang
8. TAO6 - Jl. Jawa No. 40
9. T3LU - Jl. Aceh No. 41 RT 01 RW 04
10. TST4 - Jl.Ahmad Yani No.233
11. TB79 - Jl. Jend. A. Yani No. 269 RT 01 RW 08
12. FNLZ - Jl. Bengawan
13. TL2Z - Jl. Ambon
14. Rumah Produksi Mister Donut – Mister Donut Dago

Jarak yang ditempuh dalam sekali pengiriman adalah 15.404 m atau 15,404 km. Program selesai dieksekusi dalam waktu 0,2244 detik.



Gambar 4.1 Visualisasi rute optimum dalam bentuk graf berarah dan berbobot(Sumber: Dokumen Pribadi)

Gambar 4.1 menunjukkan sirkuit Hamilton dengan bobot minimum sebagai rute optimum pengiriman produk Mister Donut. Bobot dalam graf tersebut mewakili jarak antara dua lokasi dalam meter.

B. Analisis Kompleksitas Waktu Penyelesaian TSP dengan Pemrograman Dinamis

Dalam algoritma yang diimplementasikan, terdapat $n \cdot 2^n$ sub-rute yang dihitung karena terdapat *n* buah simpul dan dicatat 2^n simpul lain yang belum dikunjungi pada tiap langkah. Setiap subrute dihitung dengan kompleksitas waktu O(*n*), sehingga kompleksitas waktu keseluruhan program adalah $O(n^2 \cdot 2^n)$. Pendekatan pemrograman dinamis menunjukkan performa yang lebih efisien dibandingkan pendekatan *brute-force* yang memiliki kompleksitas O(*n!*). Meskipun demikian, $O(n^2 \cdot 2^n)$ tetaplah kurang efisien untuk n > 16. Pendekatan pemorgraman dinamis dapat digunakan untuk persoalan penentuan rute pengiriman produk Mister Donut secara efisien karena jumlah cabang Indomaret yang dibatasi ≤ 15.

V. Kesimpulan

Perencanaan rute pengiriman produk Mister Donut ke cabang-cabang Indomaret di kota Bandung dapat dicari dengan memodelkan permasalahan tersebut sebagai Travelling Salesman Problem (TSP) yang diselesaikan dengan program yang mengimplementasikan pemrograman dinamis(*dynamic programming*). Solusi ini efisien karena setiap truk perlu mengirim ke lebih sedikit dari 15 cabang Indomaret sekali pengiriman. Rute optimum untuk shift pengiriman salah satu truk, yaitu : Mister Donut Dago → T0BX → F308 → TT11 →

TOYJ → TKUT → T4C3 → TX0M → TAO6 → T3LU → TST4 → TB79 → FNLZ → TL2Z → Mister Donut Dago dengan total jarak tempuh 15,404 km. Rute pengiriman lainnya dapat dicari dengan metodologi yang sama.

VI. Ucapan Terima Kasih

Penulis menyampaikan ucapan terima kasih kepada:

1. Tuhan Yang Maha Esa atas berkat-Nya sehingga penulis dapat menyelesaikan makalah ini dengan baik,
2. Kedua orang tua penulis yang telah memberikan dukungan kepada penulis,
3. Bapak Dr. Ir. Rinaldi Munir, M.T., Ibu Fariska Zakhralativa Ruskanda S.T.,M.T, dan Ibu Dr. Nur Ulfa Maulidevi S.T., M.Sc selaku dosen mata kuliah IF2120 Matematika Diskrit yang telah membimbing penulis dan memberikan banyak ilmu yang bermanfaat selama perkuliahan.
4. Gandhiko Arya yang telah membantu penulis mengumpulkan informasi dan data Indomaret yang dibutuhkan untuk penyusunan makalah ini.

Akhir kata, penulis mengharapkan makalah ini dapat bermanfaat bagi pihak yang membacanya.

Referensi

[1] Munir, Rinaldi. (2020). “Graf: Bagian 1”. https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian1.pdf (diakses tanggal 6 Desember 2022).

[2] Munir, Rinaldi. (2020). “Graf: Bagian 2”. https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian2.pdf (diakses tanggal 6 Desember 2022).

[3] Munir, Rinaldi. (2020). “Graf: Bagian 3”. https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2020-2021/Graf-2020-Bagian3.pdf (diakses tanggal 6 Desember 2022).

[4] A., Niels, B., Paul, and Marie Schmidt. (2017). “Dynamic Programming Approaches for the Traveling Salesman Problem With Drone”. Erasmus University Digital Repository.

[5] Halim, S., Halim, F., Skiena, S. S., & Revilla, M. A. (2013). Competitive Programming 3. Lulu Independent Publish.

[6] Rosen, Kenneth. (2012). “Discrete Mathemathic and it’s Application Seventh Edition” . McGraw-Hill International.

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, 9 Desember 2022

Rachel Gabriela Chen 13521044