Aplikasi Travelling Salesman Problem untuk Menentukan Rute Pengiriman Paket di Kota Batam

Vincent Ho - 13520093
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13520093@std.stei.itb.ac.id

***Abstract*—Seiring dengan perkembangan jaman, perkembangan teknologi dan ilmu pengetahuan telah banyak membantu manusia untuk melakukan segala sesuatu dengan lebih efektif dan efisien. Salah satu contohnya adalah pembelian barang melalui internet atau belanja *online*. Hal tersebut tentu saja meningkatkan jumlah pengiriman paket di seluruh dunia termasuk Indonesia. Dalam pengiriman paket tentu saja harus dipilih rute paling optimal agar tidak memakan banyak waktu dan sumber daya. Pada makalah ini penulis membahas mengenai salah satu permasalahan dalam teori graf yaitu *Travelling Salesman Problem* (TSP) dengan menggunakan algoritma *Branch and Bound* untuk menemukan rute pengiriman paket yang paling optimal di Kota Batam.**

***Keywords*—*Branch and Bound*, graf, Kota Batam, *Travelling Salesman Problem*.**

I. Pendahuluan

Perkembangan teknologi dan ilmu pengetahuan yang pesat menyebabkan banyak perubahan pada dunia. Saat ini dunia telah memasuki era digital dengan internet sebagai salah satu tonggak utama digitalisasi. Digitalisasi membawa dampak baik yang besar pada segala aspek kehidupan manusia. Banyak kegiatan manusia yang menjadi lebih mudah dan efisien untuk dilakukan semenjak adanya internet dan teknologi-teknologi lain yang mendukung.

Perkembangan teknologi tersebut juga berdampak besar dalam bidang perdagangan. Saat ini masyarakat dunia lebih sering membeli barang atau berbelanja secara *online* melalui platform seperti Shopee, Tokopedia, Lazada, dan lain sebagainya. Hal ini dikarenakan berbelanja melalui internet jauh lebih mudah dibandingkan berbelanja secara konvensional. Pembeli tidak harus keluar rumah untuk membeli sesuatu melainkan tinggal membuka aplikasi pada *smartphone* dan menekan beberapa tombol maka barang sudah terbeli. Harga barang di *online shop* juga cenderung lebih murah dan karena banyaknya pilihan barang maka masyarakat pun jauh lebih tertarik untuk membelinya.

Banyaknya pesanan yang masuk dari berbagai belahan dunia termasuk Indonesia, maka jasa ekspedisi atau pengiriman paket juga semakin banyak digunakan. Dalam proses pengiriman paket, terdapat banyak sekali rute yang dapat dipilih dan dilalui untuk menyampai tujuan. Tentu saja jasa pengiriman paket harus menentukan rute pengiriman yang paling pendek atau paling optimal agar tidak memakan banyak waktu, biaya, dan sumber daya. Permasalahan pemilihan rute pengiriman paket ini dapat diselesaikan dengan menggunakan pendekatan graf, yaitu penyelesaian *Travelling Salesman Problem* (TSP).

Pada makalah ini, penulis akan membahas secara spesifik mengenai aplikasi *Travelling Salesman Problem* (TSP) dalam menentukan rute pengiriman paket di Kota Batam dengan menggunakan algoritma *Branch and Bound* untuk mengoptimalkan pemilihan rute perjalanan tersebut.

II. Landasan Teori

A. Graf

1. Defenisi Graf

Graf dalam struktur diskrit digunakan untuk merepresentasikan objek-objek diskrit dan hubungan antara objek-objek tersebut. Sehingga secara sederhana graf didefinisikan sebagai kumpulan titik atau simpul (*node/vertex*) yang dihubungkan oleh garis-garis (*edge*).

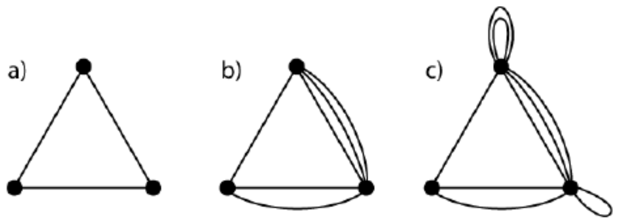
Graf G didefinisikan dengan notasi G = (V, E) yang dalam hal ini V adalah himpunan tidak kosong dari simpul-simpul (*vertices*) dan E adalah himpunan sisi (*edges*) yang menghubungkan sepasang simpul. $V = \{v_1, v_2, v_3, \ldots, v_n\}$. $E = \{e_1, e_2, \ldots, e_n\}$.

2. Jenis-Jenis Graf

Berdasarkan ada tidaknya gelang dan sisi ganda pada suatu graf, graf dapat dikelompokkan sebagai berikut:

a. Graf sederhana (*simple gr*aph)
   Graf sederhana adalah graf yang tidak mengandung gelang maupun sisi ganda.

b. Graf tak-sederhana (*unsimple-graph*)
   Graf tak-sederhana merupakan graf yang mengandung sisi ganda atau sisi gelang. Graf tak-sederhana dapat dibedakan lagi menjadi 2, yaitu:
   1. Graf ganda (*multi-graph*)
      Graf ganda merupakan graf yang mengandung sisi ganda.
   2. Graf semu (*pseudo graph*)
      Graf semu merupakan graf yang mengandung sisi gelang



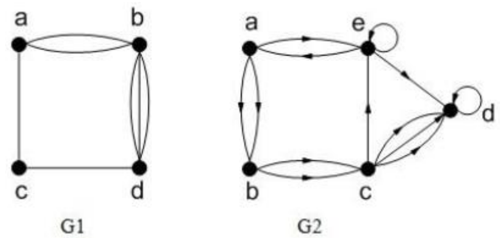
.

Gambar 1. (a) Graf sederhana. (b) Graf ganda. (c) Graf semu.
Sumber: Graf Bagian 1 oleh Rinaldi Munir

Berdasarkan ada tidaknya orientasi arah pada sisi graf, graf dibedakan menjadi 2 jenis:

a. Graf tak-berarah (*undirected graph*)
   Graf tak-berarah adalah graf yang sisinya tidak mempunyai orientasi arah.

b. Graf berarah (*directed graph* atau *digraph*)
   Graf berarah adalah graf yang setiap sisinya diberikan orientasi arah.



Gambar 2. (G1) Graf tak berarah. (G2) Graf berarah.
Sumber: Graf Bagian 1 oleh Rinaldi Munir

3. Terminologi Graf

Terdapat beberapa terminologi atau istilah yang berkaitan dengan graf. Beberapa terminologi tersebut, yaitu:

a. Ketetanggaan (*Adjacent*)
   Dua buah simpul pada graf dikatakan bertetangga jika keduanya terhubung langsung oleh sebuah sisi.

b. Bersisian (*Incidency*)
   Sebuah sisi e dikatakan bersisian dengan simpul v_j dan simpul v_k apabila terdapat sembarang e = (v_j, v_k).

c. Derajat (*Degree*)
   Derajat suatu simpul adalah jumlah sisi yang bersisian dengan simpul tersebut. Derajat ditulis dengan notasi d(v).

d. Lintasan (*Path*)
   Lintasan yang memiliki panjang n dari simpul awal v_0 ke simpul tujuan v_n di dalam graf G adalah barisan yang berselang-seling simpul-simpul dan sisi-sisi yang berbentuk v_0, e_1, v_1, e_2, v_2, …, v_{n-1}, e_n, v_n sedemikian sehingga sisi-sis dari graf G adalah $e_1 = (v_0, v_1)$, $e_2 = (v_1, v_2)$, …, $e_n = (v_{n-1}, v_n)$.

e. Sirkuit (*Circuit*)
   Sirkuit merupakan lintasan yang berawal dan berakhir pada simpul yang sama.

f. Keterhubungan (*Connected*)
   Dua buah simpul v1 dan v2 dikatakan terhubung jika terdapat lintasan dari v1 ke v2. Graf G disebut graf terhubung (*connected graph*) jika untuk setiap pasang simpul dalam himpunan V terdapat lintasan dari vi ke vj.

g. Upagraf (*Subgraph*) dan Komplemen Upagraf
   Misalkan G = (V, E) adalah sebuah graf. Maka G1 = (V1, E2) adalah upagraf (*subgraph*) dari G jika V1 $\subseteq$ V dan E1 $\subseteq$ E.
   Komplemen dari upagraf G_1 terhadap graf G adalah graf $G_2 = (V_2, E_2)$ sedemikian sehingga $E_2 = E - E_1$ dan V_2 adalah himpunan simpul yang anggota-anggota E_2 bersisian dengannya.

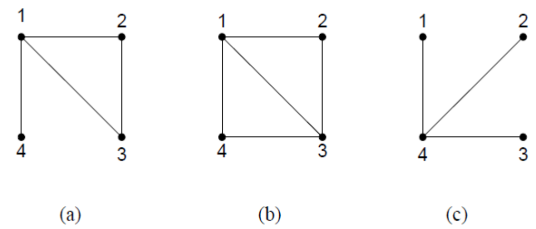
h. Upagraf merentang (*Spanning Subgraph*)
   Upagraf $G_1 = (V_1, E_1)$ dari G = (V, E) dikatakan upagraf rentang jika V_1 = V yang artinya G_1 mengandung semua simpul dari G.

i. Graf berbobot (*weighted graph*)
   Graf berbobot merupakan graf yang setiap sisinya memiliki bobot atau harga tertentu.

j. Graf lengkap (*Completed* graph)
   Graf lengkap merupakan graf sederhana yang setiap simpulnya memiliki sisi ke semua simpul lainnya. Graf lengkap dengan n buah simpul dilambangkan dengan Kn. Jumlah dari sisi pada sebuah graf lengkap yang terdiri dari n buah simpul dapat dihitung dengan rumus E = n(n-1)/2.

3. Lintasan dan Sirkuit Hamilton

Lintasan Hamilton adalah lintasan yang melalui tiap simpul di dalam graf tepat satu kali. Sirkuit Hamilton adalah sirkuit yang melalui setiap simpul di dalam graf tepat satu kali, kecuali simpul asal yang sekaligus merupakan simpul akhir yang dilalui dua kali. Graf yang memiliki sirkuit Hamilton dinamakan graf Hamilton, sedangkan graf yang hanya memiliki lintasan Hamilton disebut graf semi-Hamilton.

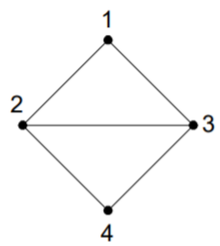


Gambar 3. (a) Graf dengan lintasan Hamilton. (b) Graf dengan sirkuit Hamilton. (c) Graf yang tidak memiliki lintasan dan sirkuit Hamilton.
Sumber: Graf Bagian 3 oleh Rinaldi Munir

4. Representasi Graf

Terdapat berbagai cara untuk merepresentasikan sebuah graf. Pada makalah ini penulis hanya akan memuat cara merepresentasikan sebuah graf menggunakan matriks ketetanggaan (*adjacency matrix*).

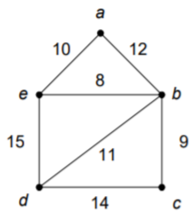
Matriks Ketetanggaan



$$\begin{array}{c c} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

Gambar 4. Matriks ketetanggaan
Sumber: Graf Bagian 2 oleh Rinaldi Munir

Misalkan matriks ketetanggaan adalah A. Maka A = [a_{ij}] dengan a_{ij} merupakan setiap elemen dari matriks tersebut. a_{ij} bernilai 1 jika simpul i dan j bertetangga dan bernilai 0 jika simpul i dan j tidak bertetangga.



$$\begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array}\begin{array}{c} \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{bmatrix} \infty & 12 & \infty & \infty & 10 \\ 12 & \infty & 9 & 11 & 8 \\ \infty & 9 & \infty & 14 & \infty \\ \infty & 11 & 14 & \infty & 15 \\ 10 & 8 & \infty & 15 & \infty \end{bmatrix} \end{array}$$

Gambar 5. Matriks ketetanggaan
Sumber: Graf Bagian 2 oleh Rinaldi Munir

Pada graf berbobot, a_{ij} diisi dengan bobot yang dimiliki oleh sisi yang menghubungkan simpul i dan simpul j. Jika simpul i dan simpul j tidak bertetangga maka diisi dengan ∞.

B. Travelling Salesman Problem (TSP)

Travelling Salesman Problem merupakan suatu permasalahan sirkuit Hamilton dalam teori graf dengan pertanyaan “jika diberikan daftar kota dan jarak antara setiap kota yang ada, apa jalur terpendek yang dapat ditempuh untuk mengunjungi setiap kota tepat sekali dan kembali lagi ke kota asal?” Dalam permasalahan ini, kota dapat dianggap sebagai simpul dan jalan yang menghubungkan antara dua buah kota sebagai sisi. Jarak antar kota dapat dinyatakan dalam bentuk bobot dari sisi yang menghubungkan kedua kota tersebut.

Persoalan ini diselesaikan dengan cara menentukan sirkuit Hamilton yang memiliki bobot paling kecil dari graf tersebut. Untuk suatu graf lengkap yang terdiri dari n kota, terdapat $\frac{(n-1)!}{2}$ sirkuit Hamilton yang berarti terdapat $\frac{(n-1)!}{2}$ rute berbeda yang dapat ditempuh untuk mengunjungi setiap kota tepat sekali dan kembali lagi ke kota asal. Terdapat banyak metode untuk menyelesaikan permasalahan TSP ini, namun penulis akan menggunakan algoritma *Branch and Bound* dalam makalah ini.

C. Algoritma Branch and Bound

Algoritma *Branch and Bound* atau biasa disebut juga dengan B&B merupakan algoritma yang biasa digunakan untuk menyelesaikan persoalan optimisasi. Optimisasi dalam hal ini berarti meminimalkan atau memaksimalkan suatu fungsi objektif, yang tidak melanggar batasan (*constraints*) persoalan. Algoritma ini bekerja dengan 2 prinsip dasar yaitu *branching* dan *bounding*. *Branching* dilakukan dengan cara menelusuri setiap cabang pencarian secara rekursif yang semakin lama semakin sempit atau sedikit. Apabila hanya melakukan *branching*, maka algoritma ini akan menjadi algoritma *bruteforce* yang akan mencoba semua kemungkinan. Oleh karena itu dilakukan juga *bounding* yaitu membatasi pencarian dengan cara menentukan nilai batasan untuk suatu penyelesaian optimal di simpul tersebut.

Algoritma *Branch and Bound* ini dapat digunakan untuk menyelesaikan permasalahan *Travelling Salesman Problem* ini dengan cara menentukan rute paling optimal yang dapat dilalui. Hal ini dilakukan dengan cara menelusuri jalan dengan bobot terkecil dalam setiap percabangan.

Langkah-langkah mengimplementasikan algoritma *Branch and Bound* ini dalam menyelesaikan permasalahan TSP adalah sebagai berikut:

a. Representasikan graf berbobot sebagai matriks ketetanggaan (*adjacency matrix*).
b. Lakukan reduksi terhadap setiap baris dan kolom pada matriks yang terbentuk sehingga terdapat paling sedikit sebuah angka nol pada setiap baris dan kolom.
c. Hitung batas bawah dari total bobot minimum graf dengan cara menjumlahkan semua elemen pengurang (pereduksi) dari semua baris dan kolom. Batas bawah ini akan menjadi *cost* dari simpul akar.
d. Misalkan A merupakan matriks tereduksi dari simpul R (simpul akar) dan S adalah anak dari simpul R sehingga sisi (R, S) pada pohon ruang status berkoresponden dengan sisi (i, j).
   Hitung *cost* pada matriks tereduksi untuk simpul S :
   - Ubah semua nilai pada baris i dan kolom j pada matriks A menjadi ∞.
   - Ubah nilai A(j,1) menjadi ∞.
   - Lakukan reduksi pada matriks yang terbentuk hingga pada setiap baris dan kolom paling sedikit mengandung sebuah angka nol.
   - Hitung *cost* pada simpul S dengan cara:
     C(S) = C(R) + A(i,j) + r
     C(S): bobot perjalanan minimum yang melalui simpul S.
     C(R): bobot perjalanan minimum yang melalui simpul R, dalam hal ini R adalah orang tua dari S.
     A(i,j): bobot sisi (i,j) pada graf yang berkoresponden dengan sisi (R,S)
     r: jumlah semua elemen pengurang pada proses pencarian matriks tereduksi dari simpul S.
   - Hasil reduksi tersebut dijadikan sebagai matriks A’
e. Ambilah jalur dengan *cost* terkecil untuk setiap simpul pada level yang sama, kemudian ulangi langkah d untuk level berikutnya. Langkah tersebut terus dilakukan hingga menelusuri seitap simpul yang ada.

III. PEMBAHASAN

A. Batasan Penelitian

Dalam penulisan makalah ini, terdapat beberapa batasan dan asumsi yang digunakan oleh penulis dalam menentukan rute perjalanan paling optimal untuk mengirimkan paket di Kota Batam, yaitu sebagai berikut:

1. Efisiensi rute hanya ditentukan berdasarkan jarak tempuh. Hal ini berarti faktor lain seperti kemacetan dan kualitas jalan diabaikan.
2. Apabila terdapat berbagai jalan yang dapat diambil untuk mencapai sebuah tempat, maka akan diambil jalan yang terpendek.
3. Pengiriman paaket diasumsikan selalu berawal dari kecamatan Batu Ampar.

B. Lokasi Penelitian

Lokasi yang digunakan sebagai tempat penelitian dalam makalah ini berasal dari kecamatan yang terdapat di Kota Batam, berikut ini beberapa kecamatan yang diambil:

1. Batu Ampar (hijau)
2. Sekupang (kuning)

3. Batam Kota (biru)
4. Sagulung (merah)
5. Sei Beduk (ungu)

Gambar 6. Gambar Kota Batam dengan penanda kecamatan yang diteliti.
Sumber: Google Map

Berikut merupakan lokasi penelitian yang telah direpresentasikan dalam bentuk matriks ketetanggaan (*adjacency matrix*). Graf merupakan graf berbobot dengan tiap sisi merepresentasikan jarak antara tiap kecamatan dalam satuan kilometer (km).

Kecamatan	1	2	3	4	5
1	∞	13.2	11.2	25.9	21.8
2	14.7	∞	13.3	13.9	21.9
3	14.3	13.3	∞	21.7	18.3
4	27.3	12.6	20.8	∞	20.5
5	23.9	21.2	16	18.9	∞

C. Aplikasi *Travelling Salesman Problem* (TSP) dengan menggunakan algoritma *Branch and Bound* dalam menentukan rute pengiriman paket paling optimal.

Langkah pertama yang harus dilakukan adalah merepresentasikan graf berbobot sebagai matriks ketetanggaan (*adjacency matrix*).

Kecamatan	1	2	3	4	5
1	∞	13.2	11.2	25.9	21.8
2	14.7	∞	13.3	13.9	21.9
3	14.3	13.3	∞	21.7	18.3
4	27.3	12.6	20.8	∞	20.5
5	23.9	21.2	16	18.9	∞

Kemudian lakukan reduksi terhadap setiap baris dan kolom pada matriks yang terbentuk sehingga terdapat paling sedikit sebuah angka nol pada setiap baris dan kolom.

- *Reduced Cost Matrix* (1) : Rute 1

Dilakukan reduksi pada setiap baris sehingga setiap baris mengandung paling sedikit satu nol.

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	13.2	11.2	25.9	21.8	11.2
2	14.7	∞	13.3	13.9	21.9	13.3
3	14.3	13.3	∞	21.7	18.3	13.3
4	27.3	12.6	20.8	∞	20.5	12.6
5	23.9	21.2	16	18.9	∞	16

Dilakukan reduksi pada setiap kolom sehingga setiap kolom mengandung paling sedikit satu nol.

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	2	0	14.7	10.6	11.2
2	1.4	∞	0	0.6	8.6	13.3
3	1	0	∞	8.4	5	13.3
4	14.7	0	8.2	∞	7.9	12.6
5	7.9	5.2	0	2.9	∞	16
elemen pereduksi	1	0	0	0.6	5	

Matriks reduksi yang terbentuk adalah matriks A

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	2	0	14.1	5.6	11.2
2	0.4	∞	0	0	3.6	13.3
3	0	0	∞	7.8	0	13.3
4	13.7	0	8.2	∞	2.9	12.6
5	6.9	5.2	0	2.3	∞	16
elemen pereduksi	1	0	0	0.6	5	**73**

C(1) = 11.2 + 13.3 + 13.3 + 12.6 + 16 + 1 + 0.6 + 5 = 73

Langkah selanjutnya adalah ubah semua nilai pada baris i dan kolom j untuk setiap rute yang ditempuh pada matriks reduksi yang terbentuk (Matriks A) menjadi ∞. Ubah nilai A(j,1) menjadi ∞. Kemudian lakukan reduksi pada matriks yang terbentuk hingga setiap baris dan kolom paling sedikit mengandung sebuah angka nol. Matriks reduksi yang memiliki *cost* terendah akan menjadi matriks A'.

- *Reduced Cost Matrix* (2): Rute 1 – 2

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	∞	∞	∞	∞	0
2	∞	∞	0	0	3.6	0
3	0	∞	∞	7.8	0	0
4	13.7	∞	8.2	∞	2.9	2.9
5	6.9	∞	0	2.3	∞	0
elemen pereduksi	0	0	0	0	0	

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	∞	∞	∞	∞	0
2	∞	∞	0	0	3.6	0
3	0	∞	∞	7.8	0	0
4	10.8	∞	5.3	∞	0	2.9
5	6.9	∞	0	2.3	∞	0
elemen pereduksi	0	0	0	0	0	**2.9**

C(2) = C(1) + A(1,2) + r = 73 + 2 + 2.9 = 77.9

- *Reduced Cost Matrix* (3): Rute 1-3

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	∞	∞	∞	∞	0
2	0.4	∞	∞	0	3.6	0
3	∞	0	∞	7.8	0	0
4	13.7	0	∞	∞	2.9	0
5	6.9	5.2	∞	2.3	∞	2.3
elemen pereduksi	0.4	0	0	0	0	

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	∞	∞	∞	∞	0
2	0	∞	∞	0	3.6	0
3	∞	0	∞	7.8	0	0
4	13.3	0	∞	∞	2.9	0
5	4.2	2.9	∞	0	∞	2.3
elemen pereduksi	0.4	0	0	0	0	**2.7**

C(3) = C(1) + A(1,3) + r = 73 + 0 + 2.7 = 75.7

- *Reduced Cost Matrix* (4): Rute 1-4

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	∞	∞	∞	∞	0
2	0.4	∞	0	∞	3.6	0
3	0	0	∞	∞	0	0
4	∞	0	8.2	∞	2.9	0
5	6.9	5.2	0	∞	∞	0
elemen pereduksi	0	0	0	0	0	**0**

C(4) = C(1) + A(1,4) + r = 73 + 14.1 + 0 = 87.1

- *Reduced Cost Matrix* (5): Rute 1-5

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	∞	∞	∞	∞	0
2	0.4	∞	0	0	∞	0
3	0	0	∞	7.8	∞	0
4	13.7	0	8.2	∞	∞	0
5	∞	5.2	0	2.3	∞	0
elemen pereduksi	0	0	0	0	0	**0**

C(5) = C(1) + A(1,5) + r = 73 + 5.6 + 0 = 78.6

Karena *Reduced Cost Matrix* (3) memiliki nilai *cost* yang paling kecil yaitu 75.7, maka matriks tersebut dijadikan patokan untuk langkah selanjutnya. Langkah yang sama dilakukan dengan *Reduced Cost Matrix* (3) menjadi matriks A yang baru yaitu matriks A'.

- *Reduced Cost Matrix* (6): Rute 1-3-2

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	∞	∞	∞	∞	0
2	∞	∞	∞	0	3.6	0
3	∞	∞	∞	∞	∞	0
4	13.3	∞	∞	∞	2.9	2.9
5	4.2	∞	∞	0	∞	0
elemen pereduksi	4.2	0	0	0	0	

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	∞	∞	∞	∞	0
2	∞	∞	∞	0	3.6	0
3	∞	∞	∞	∞	∞	0
4	6.2	∞	∞	∞	0	2.9
5	0	∞	∞	0	∞	0
elemen pereduksi	4.2	0	0	0	0	**7.1**

C(6) = C(3) + A'(3,2) + r = 75.7 + 0 + 7.1 = 82.8

- *Reduced Cost Matrix* (7): Rute 1-3-4

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	∞	∞	∞	∞	0
2	0	∞	∞	∞	3.6	0
3	∞	∞	∞	∞	∞	0
4	∞	0	∞	∞	2.9	0
5	4.2	2.9	∞	∞	∞	2.9
elemen pereduksi	0	0	0	0	2.9	

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	∞	∞	∞	∞	0
2	0	∞	∞	∞	0.7	0
3	∞	∞	∞	∞	∞	0
4	∞	0	∞	∞	0	0
5	1.3	0	∞	∞	∞	2.9
elemen pereduksi	0	0	0	0	2.9	**5.8**

C(7) = C(3) + A’ (3,4) + r = 75.7 + 7.8 + 5.8 = 89.3

- *Reduced Cost Matrix* (8): Rute 1-3-5

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	∞	∞	∞	∞	0
2	0	∞	∞	0	∞	0
3	∞	∞	∞	∞	∞	0
4	13.3	0	∞	∞	∞	0
5	∞	2.9	∞	0	∞	2.3
elemen pereduksi	0	0	0	0	0	**0**

C(8) = C(3) + A'(3,5) + r = 75.7 + 0 + 0 = 75.7

Karena *Reduced Cost Matrix* (8) memiliki nilai *cost* yang paling kecil yaitu 75.7, maka matriks tersebut dijadikan patokan untuk langkah selanjutnya. Langkah yang sama dilakukan dengan *Reduced Cost Matrix* (8) menjadi matriks A yang baru yaitu matriks A’’.

- *Reduced Cost Matrix* (9): Rute 1-3-5-2

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	∞	∞	∞	∞	0
2	∞	∞	∞	0	∞	0
3	∞	∞	∞	∞	∞	0
4	13.3	∞	∞	∞	∞	13.3
5	∞	∞	∞	∞	∞	0
elemen pereduksi	0	0	0	0	0	

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	∞	∞	∞	∞	0
2	∞	∞	∞	0	∞	0
3	∞	∞	∞	∞	∞	0
4	0	∞	∞	∞	∞	13.3
5	∞	∞	∞	∞	∞	0
elemen pereduksi	0	0	0	0	0	**13.3**

C(9) = C(8) + A’’(5,2) + r = 75.7 + 2.9 + 13.3 = 91.9

- *Reduced Cost Matrix* (10): Rute 1-3-5-4

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	∞	∞	∞	∞	0
2	0	∞	∞	∞	∞	0
3	∞	∞	∞	∞	∞	0
4	∞	0	∞	∞	∞	0
5	∞	∞	∞	∞	∞	0
elemen pereduksi	0	0	0	0	0	**0**

C(10) = C(8) + A’’(5,4) + r = 75.7 + 0 + 0 = 75.7

Karena *Reduced Cost Matrix* (10) memiliki nilai *cost* yang paling kecil yaitu 75.7, maka matriks tersebut dijadikan patokan untuk langkah selanjutnya. Langkah yang sama dilakukan dengan *Reduced Cost Matrix* (10) menjadi matriks A yang baru yaitu matriks A’’’.

- *Reduced Cost Matrix* (11): Rute 1-3-5-4-2

Kecamatan	1	2	3	4	5	elemen pereduksi
1	∞	∞	∞	∞	∞	0
2	∞	∞	∞	∞	∞	0
3	∞	∞	∞	∞	∞	0
4	∞	∞	∞	∞	∞	0
5	∞	∞	∞	∞	∞	0
elemen pereduksi	0	0	0	0	0	**0**

C(11) = C(10) + A’’’(4,2) + r = 75.7 + 0 + 0 = 75.7

D.Hasil Penelitian

Berdasarkan hasil perhitungan dan pencarian rute paling optimal dari pengiriman paket di Kota Batam menggunakan algoritma *Branch and Bound*, diperoleh rute yang paling optimal untuk persoalan ini adalah rute 1-3-5-4-2-1, yaitu Batu Ampar – Batam Kota – Sei Beduk – Sagulung – Sekupang – Batu Ampar, dengan total jarak tempuh 75.7 km.

IV. KESIMPULAN

Pemilihan rute terpendek untuk mengirimkan paket di Kota Batam dapat dilakukan dengan menggunakan pendekatan permasalahan *Travelling Salesman Problem* (TSP) dan solusi penyelesaiannya dengan menggunakan algoritma *Branch and Bound*. Dengan menggunakan algoritma *Branch and Bound* maka dapat diperoleh rute yang mengunjungi setiap titik paling sedikit sekali dan kembali ke titik awal dengan jarak paling pendek. Rute yang diperoleh adalah Batu Ampar – Batam Kota – Sei Beduk – Sagulung – Sekupang – Batu Ampar dengan total jarak tempuh 75.7 km.

V. UCAPAN TERIMA KASIH

Penulis ingin menyampaikan ucapan syukur kepada Tuhan Yang Maha Esa atas berkat dan rahmat-Nya, penulis diberikan kesehatan dan kekuatan untuk menyelesaikan penulisan makalah ini dengan tepat waktu. Penulis ingin berterima kasih kepada kedua orang tua penulis yang telah mendukun secara moral, doa, dan material. Penulis juga ingin menyampaikan terima kasih kepada semua pihak yang telah mendukung studi dan proses pembelajaran pada mata kuliah IF2120 Matematika Diskrit terutama kepada dosen pengampu mata kuliah ini yaitu, Bapak Dr. Rinaldi Munir, Ibu Harlili, M.Sc, dan Ibu Nur Ulfa Maulidevi, atas bimbingan dan ilmu yang telah diberikan selama 1 semester tahun ajaran 2021/2022 ini.

REFERENSI

[1] Munir, Rinaldi. (2006). Diktat Kuliah IF2120: Matematika Diskrit.Bandung: Program Studi Teknik Informatika Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung. Diakses tanggal 8 Desember 2021.
[2] Applegate, D. L.; Bixby, R. M.; Chvátal, V.; Cook, W. J. (2006), The Traveling Salesman Problem, ISBN 978-0-691-12993-8. Diakses tanggal 9 Desember 2021.
[3] Clausen, Jens (1999). Branch and Bound Algorithms—Principles and Examples. Diakses tanggal 9 Desember 2021.
[4] Munir, Rinaldi Graf Bagian 1. Informatika, Bandung: 2020. Diakses tanggal 8 Desember 2021.
[5] Munir, Rinaldi Graf Bagian 2. Informatika, Bandung: 2020. Diakses tanggal 8 Desember 2021.
[6] Munir, Rinaldi Graf Bagian 3. Informatika, Bandung: 2020. Diakses tanggal 8 Desember 2021.
[7] Munir, Rinaldi Algoritma Branch and Bound Bagian 1. Informatika, Bandung: 2021. Diakses tanggal 9 Desember 2021.
[8] Munir, Rinaldi Algoritma Branch and Bound Bagian 2. Informatika, Bandung: 2021. Diakses tanggal 9 Desember 2021.

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 Desember 2020

Vincent Ho (13520093)