# BAB II
# KAJIAN PUSTAKA

Pada BAB II akan dibahas beberapa teori yang diperlukan untuk pembahasan pada BAB III. Teori-teori yang akan dibahas tersebut mengenai *Graf, Vehicle Routing Problem, Capacitated Vehicle Routing Problem*, *Saving Matriks*, *Sequential Insertion,* dan *Nearest Neighbour.*

## A. Graf

### 1. Pengertian Graf

Menurut Sugeng Mardiyono (1996:1) Suatu Graf G dapat dipandang sebagai kumpulan dari titik yang disebut simpul dan segmen garis yang menghubungkan 2 simpul yang disebut rusuk. Graf dalam kehidupan sehari-hari, digunakan untuk menggambarkan berbagai macam struktur yang ada. Tujuannya adalah sebagai visualisasi objek-objek agar lebih mudah dimengerti. Himpunan simpul pada sebuah graf $G$ dinyatakan dengan $V(G)$, dimana $V(G) = \{v_1, v_2, v_3, \dots, v_n\}$. Sedangkan himpunan rusuk pada sebuah graf G dinyatakan dengan $E(G)$, dimana $E(G) = \{e_1, e_2, e_3, \dots, e_n\}$. Sehingga secara matematis, graf dapat dilambangkan dengan $= \big(V(G), E(G)\big)$ .

Jika $v_1$ dan $v_2$ adalah sepasang simpul yang berbeda di *G* dan $e = v_1, v_2$adalah sebuah rusuk di *G*, maka

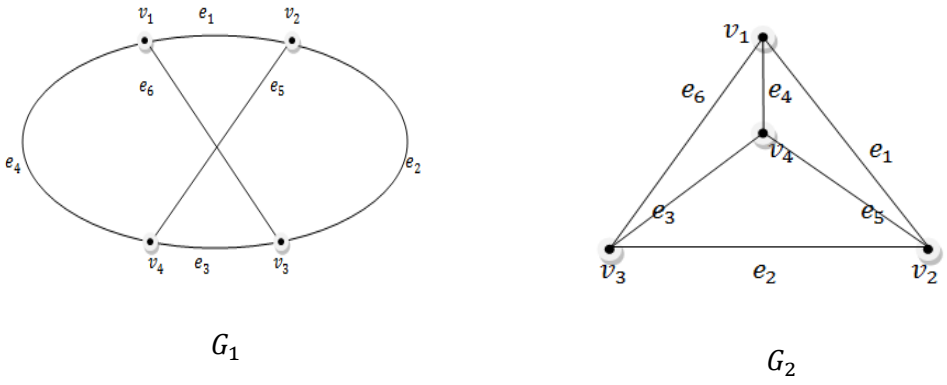
a. $v_1$dan $v_2$ merupakan simpul-simpul ujung rusuk *e* (*e* menghubungkan $v_1$ dan$v_2$ di *G*).

b. $v_1$dan $v_2$ berikatan (*adjacent*) di *G*.

c. Rusuk *e* ada (*incident*) pada simpul$v_1$dan $v_2$.

Dari Gambar 2.1 tampak bahwa graf $G_1$ digambarkan secara berbeda dengan graf $G_2$, namun pada dasarnya $G_1$ dan $G_2$ adalah dua graf yang sama karena mempunyai jumlah simpul yang sama, mempunyai jumlah rusuk yang sama, berkorespondensi satu-satu antara simpul-simpul keduanya dan antara rusuk-rusuk keduanya.

Berikut Gambar 2.1 yang menggambarkan graf $G = (V(G), E(G))$dengan $\mathrm{V(G)} = \{\mathrm{v_1}, \mathrm{v_2}, \mathrm{v_3}, \mathrm{v_4}\}$ dan $\mathrm{E(G)} = \{\mathrm{e_1}, \mathrm{e_2}, \mathrm{e_3}, \mathrm{e_4}, \mathrm{e_5}, \mathrm{e_6}\}$.



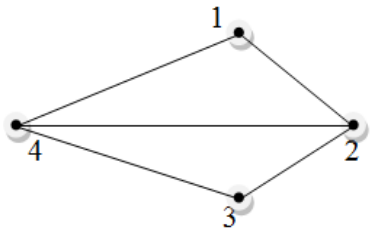

Gambar 2.1 Graf $G = (V(G), E(G))$

Menurut Liu (1995: 144-145) graf yang memiliki orientasi arah pada sisi secara umum dapat diklasifikasikan menjadi dua jenis, yaitu sebagai berikut:

a. Graf tak berarah (*undirected graph*)

Graf tak-berarah $G$ didefinisikan sebagai suatu pasangan terurut $G = (V(G), E(G))$ dengan tidak memperhatikan arah. Suatu graf tak berarah dapat direpresentasikan secara geometrik sebagai himpunan simpul-simpul $V(G)$ dengan suatu himpunan rusuk-rusuk $E(G)$ antar simpul-simpul tersebut. Graf tak berarah adalah graf yang tidak memiliki orientasi arah.

Pada Gambar 2.2 akan diberikan contoh graf tak berarah.

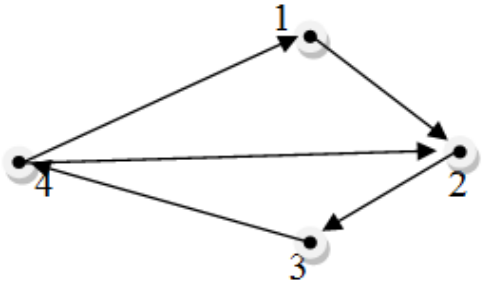



Gambar 2.2 Graf tak berarah

Pada graf tak berarah, urutan pasangan simpul yang dihubungkan oleh rusuk tidak diperhatikan. Dalam hal ini $(v_i, v_j)$ dan $(v_j, v_i)$ merupakan pasangan rusuk yang sama.Dengan kata lain $(v_i, v_j) = (v_j, v_i)$.

b. Graf berarah (*directed graph* atau *digraph*)

Graf berarah didefinisikan sebagai pasangan terurut $G = (V(G), E(G))$ dengan memperhatikan arah rusuk. Graf berarah dapat digambarkan secara geometris sebagai himpunan simpul-simpul $V(G)$ dengan himpunan tanda panah antara pasangan simpul-simpul. Graf yang setiap rusuknya diberikan orientasi arah adalah sebuah graf berarah. Pada Gambar 2.3akan diberikan contoh graf berarah.




Gambar 2.3 Graf berarah

Pada graf berarah $(v_i, v_j)$ dan $(v_j, v_i)$ menyatakan dua buah rusuk yang berbeda. Dengan kata lain $(v_i, v_j) \neq (v_j, v_i)$. Pada rusuk $(v_i, v_j)$, simpul $v_i$ dinamakan simpul awal (*initial vertex*) dan simpul$v_j$ dinamakan simpul terminal (*terminal vertex*).

**2. Macam – macam Graf**

Menurut Sugeng Mardiyono (1996:32) sesuai dengan kekhasan strukturnya, graf dapat diklasifikasikan dalam beberapa jenis yaitu graf sederhana, tidak sederhana, berarah, teratur, berbobot, pohon dan sebagainya.

a. Jenis graf berdasarkan ada tidaknya gelang dan rusuk ganda.

Berdasarkan ada tidaknya gelang dan rusuk ganda graf dapat dibedakan menjadi 2 jenis yaitu graf sederhana dan tidak sederhana. Dalam sebuah graf ada kemungkinan dijumpai dua rusuk atau lebih yang menghubungkan dua simpul yang sama. Rusuk seperti ini disebut rusuk ganda. Ada pula rusuk yang menghubungkan simpul tertentu dengan dirinya sendiri yang disebut gelang (*Loop*). Dengan demikian, berdasar ada tidaknya gelang atau rusuk ganda, graf dapat dibedakan menjadi 2 jenis.

1) Graf sederhana

Graf sederhana adalah graf yang tidak memuat rusuk ganda dan gelang. Beberapa graf sederhana dapat ditunjukkan sebagai berikut :

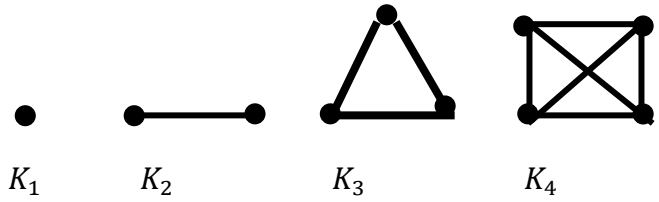
i. Graf nol adalah graf yang tidak memiliki rusuk atau himpunan rusuknya merupakan himpunan kosong. Gambar berikut menunjukkan graf nol dengan 2 buah simpul.

Contoh 2.4 :


Gambar 2.4 Graf nol dengan banyak simpul 2

ii. Graf lengkap adalah graf sederhana yang setiap pasang simpulnya saling berikatan. Notasi graf lengkap n simpul adalah $K_n$. Setiap simpul pada Kn berderajat $n-1$. Jumlah rusuk pada graf lengkap yang terdiri dari n buah simpuladalah $n\ (n-1)/2$.

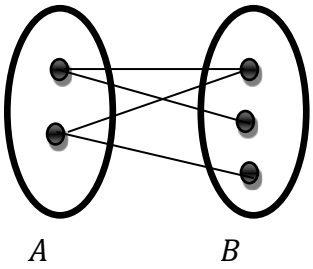
Contoh 2.5 :




Gambar 2.5 Graf lengkap

iii. Graf bipartit adalah graf sederhana yang himpunan simpulnya dapat dibagi menjadi 2 buah himpunan bagian$A$ dan $B$. Sedemikian sehingga setiap rusuk pada graf tersebut menghubungkan sebuah simpul di $A$ ke sebuah simpul di $B$

Contoh 2.6:




Gambar 2.6 Graf bipartit

2) Graf tidak sederhana

Graf tidak sederhana adalah graf yang memiliki gelang atau rusuk ganda.

Contoh 2.7 :


Gambar 2.7 Graf tidak sederhana

b. Jenis graf berdasarkan keteraturan derajat simpulnya

Berdasarkan keteraturan derajat dari simpulnya graf dapat dibedakan menjadi 2 jenis yaitu :

1) Graf teratur

Graf teratur adalah graf yang setiap simpulnya berderajat sama. Derajat suatu simpul pada graf adalah jumlah rusuk yang bersisian dengan simpul tersebut. Sedangkan jumlah rusuk pada graf teratur adalah nr/2, dimana r adalah derajat graf teratur dan n adalah banyaknya simpul.

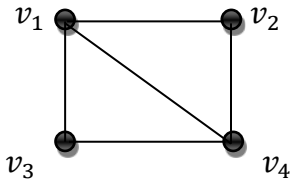
Contoh 2.8 :


Gambar 2.8 Graf teratur

2) Graf tidak teratur

Graf tidak teratur adalah graf yang tidak setiap simpulnya mempunyai derajatyang sama.

Contoh 2.9 :

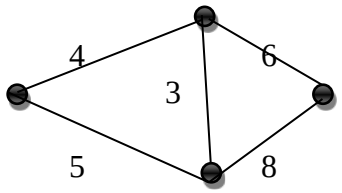



Gambar 2.9. Graf tidak teratur

c. Graf Berbobot (*weighted graph*)

Graf yang setiap rusuknya diberi sebuah harga (bobot).

Contoh 2.10 :




Gambar 2.10 Graf Berbobot

**3. Keterhubungan**

a. Jalan, jejak, lintasan, sirkuit dan sikel

Pada bagian ini akan dijelaskan pengertian jalan, jejak, lintasan, sirkuit dan sikel yang disertai dengan contoh untuk memperjelas definisi tersebut.

1) Jalan (*walk*) pada graf $G$didefinisikan sebagai barisan berhingga (tak kosong).

w = $(v_{0,}e_1, v_1, e_2, \ldots., e_n, v_n)$ dengan $v_0$disebut simpul awal dan $v_n$ simpul akhir, yang suku-sukunya bergantian simpul dan rusuk sedemikian hingga ujung$e_i$ adalah $v_{i-1}$dan $v_i$ adalah simpul-simpul akhir rusuk $e_i$untuk $1 \leq i \leq n$. Jalan disebut tertutup jika simpul awal dan simpul akhirnya berimpit.

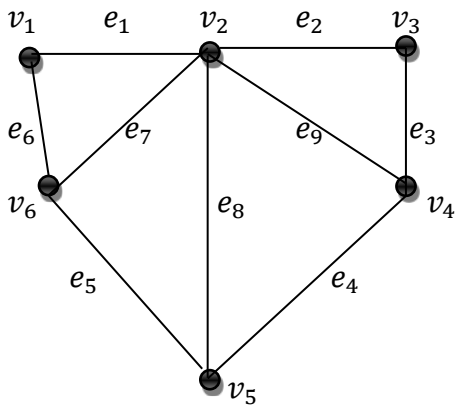
2) Jejak (*trail*) merupakan jalan tanpa rusuk berulang

3) Lintasan (*path*) merupakan jalan tanpa simpul dan rusuk berulang

4) Sirkuit (*circuit*) merupakan jejak yang tertutup

5) Sikel (*cycle*) merupakan lintasan yang simpul awal dan simpul akhirnya berimpit. Jika sikel tersebut memuat semua simpul dalam graf $G$maka disebut sikel Hamilton dan graf yang memuat sikel Hamilton disebut graf Hamilton.

Lebih jelasnya diperhatikan gambar berikut




Gambar 2.11. Graf $G_1$

i. $w_1 = (v_1, e_1, v_2, e_2, v_3, e_3, v_4, e_3, v_3)$adalah sebuah jalan di $G_1$yang panjangnya 4, karena dalam barisan ini rusuk $e_3$ muncul lebih dari sekali maka $w_1$ bukan jejak.

ii. $w_2 = (v_1, e_1, v_2, e_9, v_4, e_4, v_5, e_8, v_2)$ adalah sebuah jejak di $G_1$ yang panjangnya 4, karena dalam barisan ini simpul $v_2$muncul lebih dari sekali maka $w_2$ bukan lintasan.

iii. $w_3 = (v_1, e_6, v_6, e_5, v_5, e_4, v_4, e_3, v_3)$ adalah sebuah lintasan di $G_1$yang panjangnya 4, karena dalam barisan ini tidak ada rusuk dan simpul yang muncul lebih dari sekali.

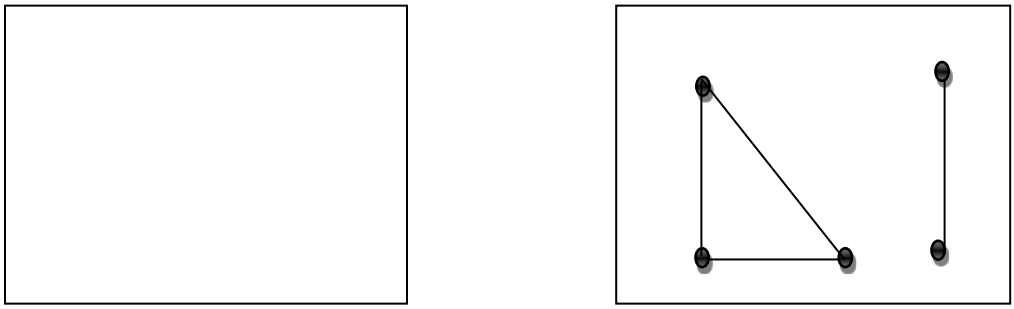
iv. $w_4 = (v_1, e_1, v_2, e_9, v_4, e_4, v_5, e_8, v_2, e_7, v_6, e_6, v_1)$adalah sebuah sirkuit di $G_1$ yang panjangnya 6, karena internal $v_2$ muncul lebih dari sekali maka $w_4$ bukan sikel.

v. $w_5 = (v_1, e_1, v_2, e_8, v_5, e_5, v_6, e_6, v_1)$adalah sebuah sikel di $G_1$yang panjangnya 4.

b. Keterhubungan graf

Suatu graf $G$ dikatakan terhubung atau *connecte*d. Jika untuk setiap dua rusuk$u$ dan $v$di $G$, terdapat lintasan yang menghubungkan simpul dan rusuk itu, sebaliknya, graf dikatakan tidak terhubung (*disconnected*) jika tidak ada lintasan yang menghubungkannya ( Mahmudi, 2001:19). Jika suatu graf tidak terhubung maka graf $G$ akan terdiri beberapa subgraf yang disebut komponen graf. Banyaknya komponen graf $G$ dinotasikan dengan $\omega(G)$.Graf terhubung mempunyai satu komponen dan graf tidak terhubung mempunyai lebih dari satu komponen ( SugengMardiyono, 1996:44).

Contoh graf terhubung dan tidak terhubung pada gambar 2.13.



$G_1$ $G_2$

Gambar 2.12 Graf terhubung $G_1$ dengan satu komponen dan graf tidak terhubung $G_2$ dengan dua komponen

4. Pengertian jaringan

Menurut Kershenbaum (1993:112) sebuah graf dapat disebut sebagai sebuah jaringan jika simpul dan rusuknya dapat dikaitkan dengan bobot tertentu seperti panjang, kapasitas dan lain sebagainya.

## B. *Vehicle Routing problem* (VRP)

### 1. Pengertian *Vehicle Routing Problem* (VRP)

*Vehicle Routing Problem* pertama kali diperkenalkan oleh Damtzig dan Ramser pada tahun 1959. VRP sebenarnya merupakan perkembangan atau perluasan dari *Traveling Salesman Problem* (TSP). VRP merupakan permasalahan yang membahas mengenai pencarian rute suatu kendaraan dengan tujuan tertentu. Menurut Toth & Vigo (2002), VRP adalah masalah penentuan rute kendaraan dalam mendistribusikan barang dari tempat produksi yang dinamakan depot ke pelanggan dengan tujuan meminimumkan total jarak tempuh kendaraan. Selain dapat meminimumkan jarak tempuh kendaraan, VRP juga bertujuan meminimumkan biaya transportasi dan waktu tempuh kendaraan yang digunakan. Permasalahan VRP ini berkaitan erat dengan pendistribusian produk atau barang antara depot dan pelanggan.Depot sebagai gudang atau tempat keluar dan masuknya kendaraan yang digunakan untuk mendistribusikan produk atau barang kepada pelanggan.

Pada tahun 1964, Clarke dan Wright melanjutkan penelitian tersebut dengan memperkenalkan istilah depot sebagai tempat keberangkatan dan kembalinya kendaraan. Semenjak saat itu penelitian tentang VRP terus berkembang dalam dunia perindustrian, khususnya dalam penentuan rute pendistribusian barang .

Klasifikasi VRP bergantung pada tujuan dan pembatas yang digunakan, pembatas yang digunakan adalah waktu dan jarak sedangkan tujuan dari VRP sendiri yaitu meminimasi biaya, waktu dan jarak. Komponen-komponen yang berkaitan dalam VRP yaitu

a. Jaringan Jalan

Jaringan jalan biasanya dideskripsikan dalam sebuah graf yang terdiri dari *edge* (rusuk) yang merepresentasikan bagian jalan yang digunakan dan *vertex* (simpul) yang merepresentasikan produsen dan depot.

b. Pelanggan

Dalam menyelesaikan masalah VRP, terlebih dahulu harus menetapkan lokasi semua pelanggan yang ada. Kemudian diperhatikan pula permintaan yang dibutuhkan pelanggan tersebut. Banyaknya permintaan yang dibutuhkan oleh pelanggan sangat mempengaruhi lamanya waktu bongkar-muat *(loading-unloading)* barang. Selain itu diperhatikan juga apakah ada rentang waktu yang diisyaratkan dalam melayani pelanggan-pelanggan tersebut.

c. Depot

Lokasi dimana depot berada juga merupakan komponen yang penting, karena depot merupakan tempat awal dan berakhirnya suatu kendaraan dalam mendistribusikan barang atau produk. Kemudian perlu diketahui juga jumlah kendaraan yang ada pada depot serta jam operasional yang ditentukan pada depot. Tujuannya untuk membatasi waktu kinerja kendaraan dalam proses distribusi.

d. Pengemudi

Pengemudi memiliki kendala jam kerja harian, durasi maksimum perjalanan, dan tambahan jam lembur jika diperlukan.

e. Kendaraan

Komponen yang perlu diperhatikan dari kendaraan yaitu antara lain, jumlah dan kapasitas kendaraan yang digunakan. Kapasitas kendaraan tersebut membatasi permintaan pelanggan, artinya jumlah permintaan pelanggan tidak boleh melebihi kapasitas kendaraan yang digunakan. Kemudian ditentukan pula bahwa dalam satu rute hanya dilayani oleh satu kendaraan. Kemudian dalam satu kendaraan, disediakan alat untuk melayani pelanggan salah satunya galon dan dana yang berhubungan dengan penggunaan kendaraan tersebut, seperti misalnya bahan bakar yang dikeluarkan dan lainnya.

**2. Klasifikasi VRP**

Berbeda dengan *CVRP*, pada *VRP* jumlah kendaraannya dapat lebih dari satu. Dengan demikian melayani lebih dari 1 pelanggan dalam waktu bersamaan (s*plit delivery)* dapat dilakukan sedangkan mengambil barang untuk memenuhi permintaan pelanggan yang lain (*reloading)* dapat dihindari. Dalam perkembangan selanjutnya, *VRP* mempunyai cukup banyak variasi, antara lain :

a. *Vehicle Routing Problem with Time Windows* (*VRPTW*)

Setiap pelanggan yang dilayani oleh kendaraan memiliki waktu pelayanan

b. *Vehicle Routing Problem with Pickup and Delivery* (VRPPD)

Terdapat sejumlah barang yang perlu dipindahkan dari lokasi penjemputan tertentu ke lokasi pengiriman lainnya.

c. *Period Vehicle Routing Problem* (PVRP)

Adanya perencanaan yang berlaku untuk satuan waktu tertentu

d. *Capacitated Vehicle Routing Problem* (CVRP)

Kendaraan yang memiliki keterbatasan daya angkut (kapasitas) barang yang harus diantarkan ke suatu tempat

e. *Multi Depot VRP*

Depot awal untuk melayani pelanggan lebih dari satu.

## C. *Capacitated Vehicle Routing Problem* (CVRP)

Versi yang paling dasar dari *VRP* adalah *Capacitated Vehicle Routing Problem* (*CVRP*) yang dapat dijelaskan sebagai berikut :

1. Suatu depot harus melayani *n* pelanggan.
2. Depot mempunyai satu kendaraan dengan kapasitas tertentu $Q$ untuk melayani semua pelanggan.
3. Tiap pelanggan mempunyai permintaan sebesar $d_i$ yang harus dipenuhi dalam sekali pelayanan.
4. Karena depot hanya mempunyai satu kendaraan dengan kapasitas terbatas, maka kendaraan tersebut harus secara periodik kembali ke depot untuk mengambil barang untuk memenuhi permintaan pelanggan yang lain (*reloading*).
5. Tidak mungkin melayani lebih dari 1 pelanggan dalam waktu yang bersamaan (*split delivery*)
6. Solusi dari *CVRP* adalah sekumpulan rute yang dilalui kendaraan, dimana tiap pelanggan hanya dikunjungi sekali saja.

Pada dasarnya, dalam CVRP kendaraan akan memulai perjalanan dari depot untuk melakukan pengiriman ke masing-masing pelanggan dan akan kembali ke depot. Diasumsikan jarak atau biaya perjalanan antara semua lokasi telah diketahui. Jarak antara

dua lokasi adalah simetris, yang berarti jarak dari lokasi A ke lokasi B sama dengan jarak dari lokasi B ke lokasi A.

Tonci Caric and Hrvoje Gold, (2008) mendefinisikan CVRP sebagai suatu graf berarah $G = (V, E)$ dengan $V = \{v_0, v_1, v_2, \ldots, v_n, v_{n+1}\}$ adalah himpunan pelanggan,$v_0$menyatakan depot dan $v_{n+1}$ merupakan depot semu dari $v_0$yaitu tempat kendaraan memulai dan mengakhiri rute perjalanan. Sedangkan E =$\{v_i, v_j)$: $v_i, v_j \in V, i \neq j\}$ adalah himpunan rusuk berarah (arc) yang merupakan himpunan sisi yang menghubungkan antarsimpul. Setiap simpul $v_i \in V$ memiliki permintaan (*demand*) sebesar $d_i$ dengan $d_i$ adalah integer positif. Kapasitas identik yaitu $Q$, sehingga panjang setiap rute dibatasi oleh kapasitas kendaraan. Setiap pelanggan$(v_i, v_j)$ memiliki jarak tempuh $C_{ij}$yaitu jarak dari pelanggan $i$ ke pelanggan $j$. Jarak perjalanan ini diasumsikan simetrik yaitu $C_{ij} = C_{ji}$dan $C_{ii} = 0$.

Permasalahan dari CVRP adalah menentukan himpunan dari *K* rute kendaraan yang memenuhi kondisi berikut :

1. Setiap rute berawal dan berakhir di depot
2. Setiap pelanggan harus dilayani tepat satu kali oleh satu kendaraan
3. Total permintaan pelanggan dari setiap rute tidak melebihi kapasitas kendaraan
4. Total jarak dari semua rute diminimumkan.

Permasalahan tersebut kemudian diformulasikan ke dalam model matematika dengan tujuan meminimumkan total jarak tempuh perjalanan kendaraan. Pemodelan untuk CVRP memiliki parameter-parameter sebagai berikut :

*V* himpunan pelanggan,

*E* himpunan rusuk berarah (arc), $\{(v_i, v_j)$:$v_i, v_j \in i \neq j\}$,

$C_{ij}$ jarak antara pelanggan$v_i$ ke pelanggan $v_j$,

$d_i$ jumlah permintaan pada pelanggan $v_i$ dan

$Q$ kapasitas kendaraan

Didefinisikan variabel keputusannya adalah

$$X_{ij} = \begin{cases} 1, \text{jika} & \text{ada perjalanan dari pelanggan } v_i \text{ ke pelanggan } v_j \\ 0, & \text{jika selainnya} \end{cases}$$

Selanjutnya fungsi tujuannya meminimumkan total jarak tempuh perjalanan kendaraan. Jika z adalah fungsi tujuan, maka

Minimumkan

$$z = \sum_{i \in V} \sum_{j \in V} C_{ij} X_{ij} \qquad (2.1)$$

dengan kendala :

$$\sum_{j \in V, i \neq j} X_{ij} = 1, \qquad \forall i \in V \qquad (2.2)$$

$$\sum_{i \in V} d_i \sum_{j \in V, j \neq i} X_{ij} \leq Q \qquad (2.3)$$

$$\sum_{j \in V} X_{oj} = 1 \qquad (2.4)$$

$$\sum_{i \in V} X_{i,n+1} = 1 \qquad (2.5)$$

$$\sum_{i \in V} X_{i,j} - \sum_{j \in V} X_{i,j} = 0, \forall i, j \in V \qquad (2.6)$$

$$X_{ij} \in \{0,1\}, \forall i, j \in V, i \neq j \qquad (2.7)$$

Persamaan (2.2) menjamin setiap pelanggan hanya dikunjungi satu kali oleh satu kendaraan. Jika $x_{ij}$ bernilai 1, artinya ada perjalanan dari pelanggan $v_i$ke $v_j$. Sebaliknya jika $x_{ij}$ bernilai 0, artinya tidak ada perjalanan dari pelanggan$v_i$ke $v_j$, sehingga dapat dikatakan bahwa variabel $x_{ij}$ saling berhubungan. Persamaan (2.3) menjamin setiap kendaraan tidak melebihi kapasitas kendaraan untuk memenuhi total permintaan dalam satu rute.Muatan kendaraan untuk memenuhi permintaan pelanggan harus dimaksimalkan namun tidak lebih dari kapasitas kendaraan. Persamaan (2.4) menjamin setiap rute perjalanan kendaraan berawal dari depot. Persamaan (2.5) menjamin setiap rute perjalanan kendaraan berakhir di depot. Persamaan (2.6) menjamin kekontinuan rute, artinya kendaraan yang mengunjungi suatu pelanggan, setelah selesai melayani akan meninggalkan pelanggan tersebut. Persamaan (2.7) variabel keputusan merupakan anggota dari $\{0,1\}$ atau *integer biner.*

Menggunakan formulasi model matematis CVRP tidak terdapat subrute pada rute-rute yang terbentuk yang dikaitkan dengan batasan kapasitas kendaraan. Variabel keputusan hanya akan terdefinisi jika jumlah permintaan pelanggan$v_i$ dan pelanggan$v_j$ tidak melebihi kapasitas kendaraan.

Berdasarkan Definisi CVRP, diperoleh suatu kesimpulan mengenai input permasalahan CVRP sebagai berikut :

1. Input permasalahan CVRP adalah daftar jarak pelanggan, daftar permintaan tiap pelanggan dan kapasitas kendaraan.
2. Dalam Terminologi graf, kumpulan pelanggan atau titik permasalahan CVRP adalah sebuah graf lengkap dengan bobot rusuk adalah jarak antar pelanggan.

## D. Metode *Saving Matrix*

Rand (2009), mendefinisikan metode *Saving Matriks* adalah metode yang digunakan untuk menentukan rute distribusi produk ke wilayah pemasaran dengan cara menentukan rute distribusi yang harus dilalui dan jumlah kendaran berdasarkan kapasitas dari kendaraan tersebut agar diperoleh rute terpendek dan biaya transportasi yang minimal. Metode *Saving Matriks* juga merupakan salah satu tehnik yang digunakan untuk menjadwalkan sejumlah kendaraan terbatas dari fasilitas yang memiliki kapasitas maksimum yang berlainan. Pada metode *saving matriks* terdapat langkah-langkah atau beberapa algoritma yang harus dilakukan.

Langkah-Langkah menyelesaikan dengan metode *Saving Matriks* :

1. Langkah 1 : Menentukan matriks jarak

   Matriks jarak menyatakan jarak diantara tiap pasangan pelanggan yang harus dikunjungi. Menentukan jarak dapat menggunakan aplikasi *google earth*, *google maps*, maupun manual perhitungan dengan *speedometer* pada kendaraan yang digunakan.

Tabel 2.1 Bentuk Umum Matriks Jarak

| | $v_0$ | . . . | $v_i$ | . . . | $v_j$ | . . . | $v_n$ |
|---|---|---|---|---|---|---|---|
| $v_0$ | 0 | | | | | | |
| . . . | | 0 | | | | | |
| $v_i$ | $C_{0i}$ | | 0 | | | | |
| . . . | | | | 0 | | | |
| $v_j$ | $C_{0j}$ | | $C_{ij}$ | | 0 | | |
| . . . | | | | | | 0 | |
| $v_n$ | $C_{0n}$ | | $C_{in}$ | | $C_{jn}$ | | 0 |

2. Langkah 2 : Menentukan *Matriks* Penghematan

*Matriks* penghematan menunjukkan penghematan yang terjadi jika menggabungkan dua pelanggan. Jika *S*(*x*, *y*) menyatakan jarak yang dihemat, misalkan perjalanan dari pusat atau titik awal dalam satu rute dengan cara (jarak dari depot ke pelanggan 1 dan dari pelanggan 1 balik ke depot ditambah dengan jarak dari depot ke pelanggan 2 dan kemudian balik ke depot) dikurangi (jarak dari depot ke pelanggan 1 ke pelanggan 2 ditambah jarak dari pelanggan 2 ke depot), maka persamaan untuk mencari besarnya penghematan (*saving*) :

$$\boldsymbol{S(x, y) = Dist(Pusat, x) + Dist(Pusat, y) - Dist(x, y)} \quad \textbf{(2.8)}$$

Tabel 2.2 Bentuk umum Matriks Penghematan

| | $v_1$ | . . . | $v_i$ | . . . | $v_j$ | . . . | $v_n$ |
|---|---|---|---|---|---|---|---|
| $v_1$ | - | | | | | | |
| . . . | | - | | | | | |
| $v_i$ | $S_{ii}$ | | - | | | | |
| . . . | | | | - | | | |
| $v_j$ | $S_{ij}$ | | $S_{ij}$ | | - | | |
| . . . | | | | | | - | |
| $v_n$ | $S_{in}$ | | $S_{in}$ | | $S_{jn}$ | | - |

3. Langkah 3 : Menggabungkan pelanggan dalam rute perjalanan kendaraan

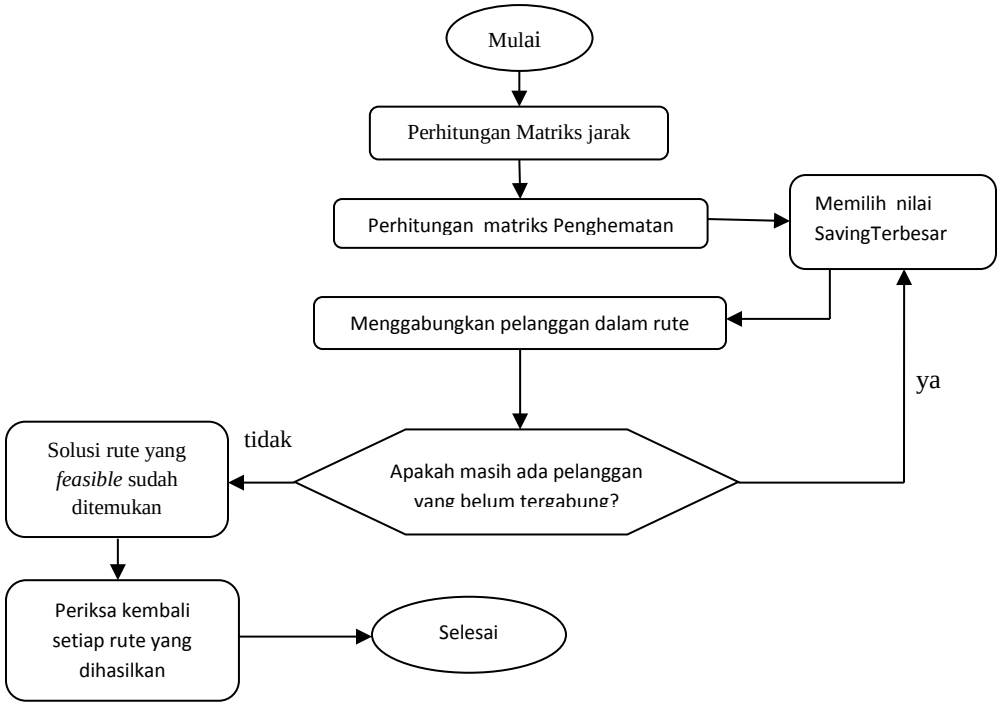
Pada tahap ini, dilakukan pembagian pelanggan kedalam suatu rute perjalanan kendaraan dengan mempertimbangkan pelanggan dan kapasitas kendaraan yang digunakan. Sebuah rute dikatakan *feasible* apabila jumlah permintaan total dari semua pelanggan tidak

melebihi kapasitas kendaraan dan jumlah permintaan dari satu pelanggan dapat ditampung secara keseluruhan oleh satu kendaraan. Prosedur yang digunakan untuk pengelompokan pelanggan yaitu berdasarkan nilai *saving matriks* terbesar. Jadi pertama-tama mengurutkan nilai *saving matriks* yang terbesar sampai kapasitas kendaraan yang digunakan dapat menampung semua permintaan. Apabila kapasitas sudah maksimal, maka prosedur tersebut akan berulang sampai semua pelanggan teralokasi dalam suatu rute perjalanan.

4. Langkah 4 : Menentukan urutan pelanggan

   Pengurutan pelanggan menggunakan prosedur *farthest insert*, *nearest insert*, dan *Nearest Neighbour*. Kemudian hasil dari ketiga prosedur tersebut dipilih mana yang menghasilkan jarak yang minimum.

   Algoritma metode *Saving matriks* sebagai berikut :



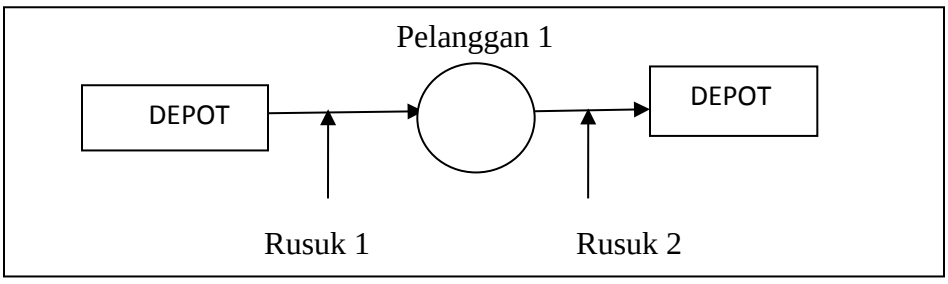

Gambar 2.13 Diagram alir metode *Saving Matriks*

## B. Metode *Sequential Insertion*

Chairul, dkk (2014) mendefinisikan metode *Sequential Insertion* sebagai metode untuk memecahkan masalah dengan cara menyisipkan pelanggan diantara pelanggan yang telah terbentuk agar didapatkan hasil yang maksimal. Untuk menjelaskan algoritma *insertion* dasar, notasi-notasi yang digunakan didefinisikan terlebih dahulu. Misal, terdapat n pelanggan dan permintaan pelanggan $i$ dinyatakan dengan $d_i$, Campbell dan Savelsberg mengasumsikan bahwa $d_i$ tidak melebihi kapasitas kendaraan Q dengan kendaraan memiliki kapasitas yang sama (*homogeneous*) (Mustika, 2008).

Sebuah rute didefinisikan sebagai perjalanan dari depot ke beberapa pelanggan secara berurutan dan kembali lagi ke depot. Sebuah rute dinyatakan dengan$V = \{0,1,2,\dots,j,\dots,n+1)$ dengan 0 dan $n+1$ menyatakan depot dan kendaraan akan melayani pelanggan $i$ yang telah menduduki posisi $j$ pada rute tersebut. Algoritma *Insertion* didefinisikan sebagai metode untuk menyisipkan pelanggan yang belum masuk dalam rute, pelanggan $i$, di antara pelanggan $j-1$ dan $j$ pada rute $V = (0,1,2,\dots,j-1,j,\dots,n+1)$.

Pada gambar 2.14 Pelanggan berikutnya dicoba untuk disisipkan pada rusuk 1 dan rusuk 2 yang ada pada rute saat ini :




Gambar 2.14. Penyisipan Pelanggan Pada Rute Saat ini

Menurut Imawati D, Suprayogi dan Yusuf P. (2008) menyatakan bahwa terdapat beberapa kriteria dalam pemilihan pelanggan pertama.Adapun kriteria pemilihan pelanggan pertama yang dapat digunakan yaitu *earliest deadline, earliest ready time, shortest time window,* dan *longest travel time.*

Adapun langkah-langkah pemecahan masalah S*equential Insertion* adalah sebagai berikut:

1. Langkah 1

   Pilih satu titik awal sebagai titik awal (depot) yang dipilih berdasarkan jarak terpendek dari depot menuju ke pelanggan pertama kembali ke depot .Selanjutnya ke langkah 2.

2. Langkah 2

   Hitung jarak tempuh yang dilalui depot ke tiap pelanggan dalam mengirimkan barang ke tiap pelanggan, lanjut ke langkah 3.

3. Langkah 3

   Hitung sisa kapasitas kendaraan, jika sisa kapasitas kendaraan memenuhi untuk mengirimkan barang sesuai permintaan pelanggan maka lanjut ke langkah 4, jika tidak lanjut ke langkah 9.

4. Langkah 4

   Jika telah memasuki pelanggan ke-2 atau seterusnya maka lanjut ke langkah 5, jika tidak lanjut ke langkah 6.

5. Langkah 5

   Sisipkan pelanggan berikutnya ke dalam urutan rute yang telah terbentuk, lanjut ke langkah 6.

6.Langkah 6

Pilih pelanggan yang memiliki jarak paling pendek, lanjut langkah 7.

7. Langkah 7

Hitung jarak tur (perjalanan), dan list rute pelanggan yang telah dilayani.Lanjut ke langkah 8.

8. Langkah 8

Jika permintaan barang yang akan dikirimkan ke pelanggan belum semua terpenuhi maka lanjut ke langkah 2, jika sudah lanjut ke langkah 10.

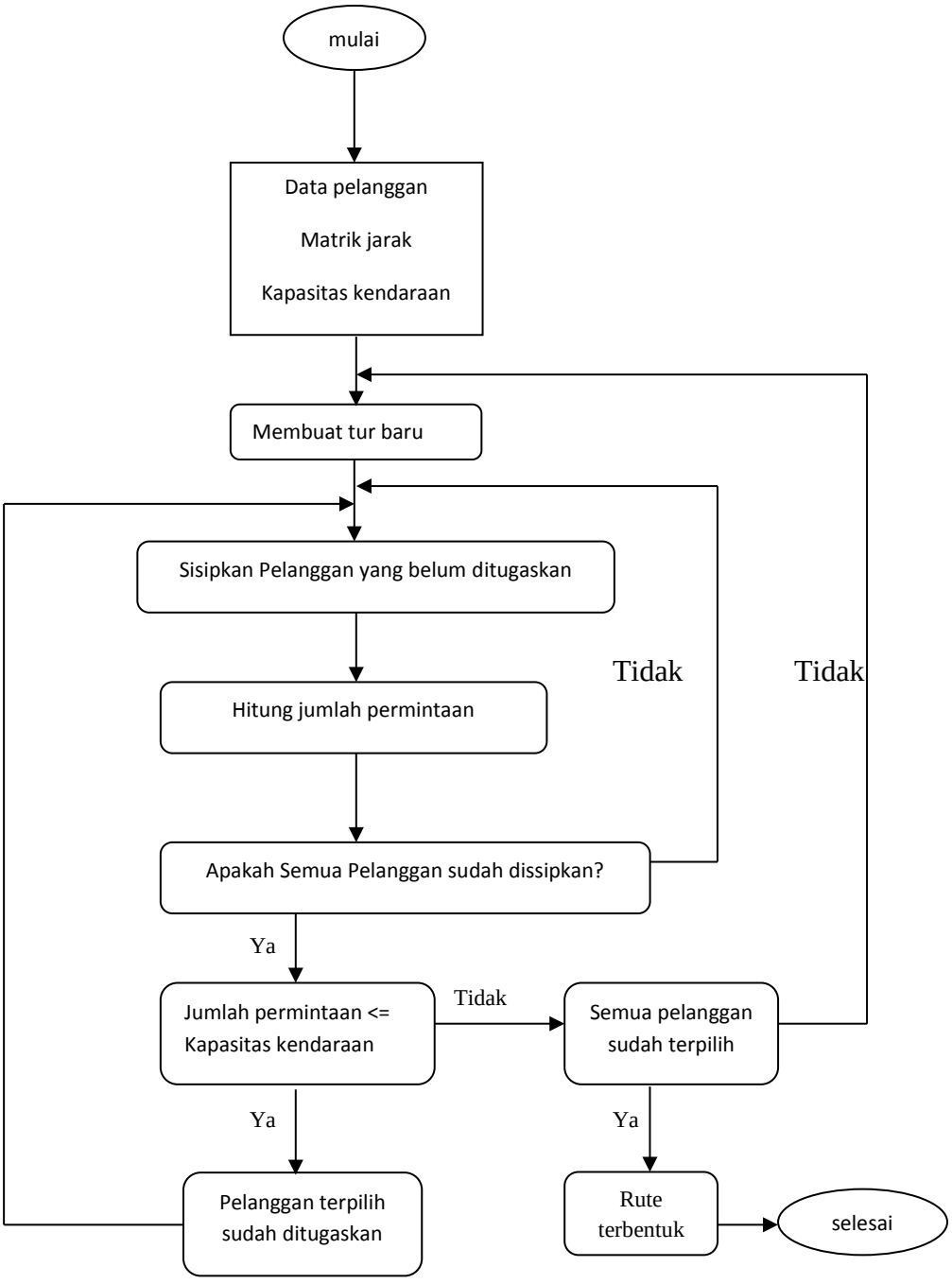
9. Langkah 9

Kembali ke depot, buat tur baru, kembali ke langkah 2.

10. Langkah 10

Semua permintaan barang yang dikirimkan ke pelanggan telah terpenuhi, hentikan prosedur ini.

Berikut diagram alir metode *Sequential Insertion* :




Gambar 2.15 Diagram Alir metode *Sequential Insertion*.

Satria Megantara, dkk (2014)

## C. Metode *Nearest Neighbour*

Metode *Nearest Neighbour* pertama kali diperkenalkan pada tahun 1983 dan merupakan metode yang sangat sederhana. Pada setiap iterasinya, dilakukan pencarian pelanggan terdekat dengan pelanggan yang terakhir untuk ditambahkan pada akhir rute tersebut. Rute baru dimulai dengan cara yang sama jika tidak terdapat posisi yang fisibel untuk menempatkan pelanggan baru karena kendala kapasitas atau *time* windows (Braysy & Gendreau, 2005). Cara kerja metode ini adalah pertama-tama, semua rute kendaraan masih kosong. Dimulai dari rute kendaraan pertama, metode ini memasukkan (*insert*) satu persatu pelanggan terdekat (*Nearest Neighbour*) yang belum dikunjungi ke dalam rute, selama memasukkan pelanggan tersebut ke dalam rute kendaraan tidak melanggar batasan kapasitas maksimum kendaraan tersebut. Kemudian proses yang sama juga dilakukan untuk kendaraan-kendaraan berikutnya, sampai semua kendaraan telahpenuh atau semua pelanggan telah dikunjungi (Gunawan, 2012).

Kelebihan dari algoritma dari *Nearest Neighbour* adalah memiliki iterasi pendek sehingga memberikan hasil optimal untuk menyelesaikan permasalahan optimasi kombinatorial. Oleh karena itu beberapa perjalanan yang menggunakan metode *Nearest Neighbour* dapat dijadikan sebagai rute awal yang dapat menghasilkan perbaikan bagi metode lain.

Kelemahan dari algoritma dari *Nearest Neighbour* adalah disaat titik tersebut mencapai titik lebih dari 20 maka proses perhitungannya cukup lama sehingga mengusahakan suatu cara untuk mencari hasil yang baik bukan yang terbaik. Namun demikian, beberapa kota yang tidak terlalu jauh dapat dilewati dan kemudian dikunjungi

pada saat akhir yang akibatnya jaraknya berubah menjadi lebih jauh dan biayanya lebih mahal.

Langkah-langkah metode *Nearest Neighbour* (Pop, 2011) adalah sebagai berikut :

1. Langkah 1

   Berawal dari depot, kemudian mencari pelanggan yang belum dikunjungiyang memiliki jarak terpendek dari depot sebagai lokasi pertama.

2. Langkah 2

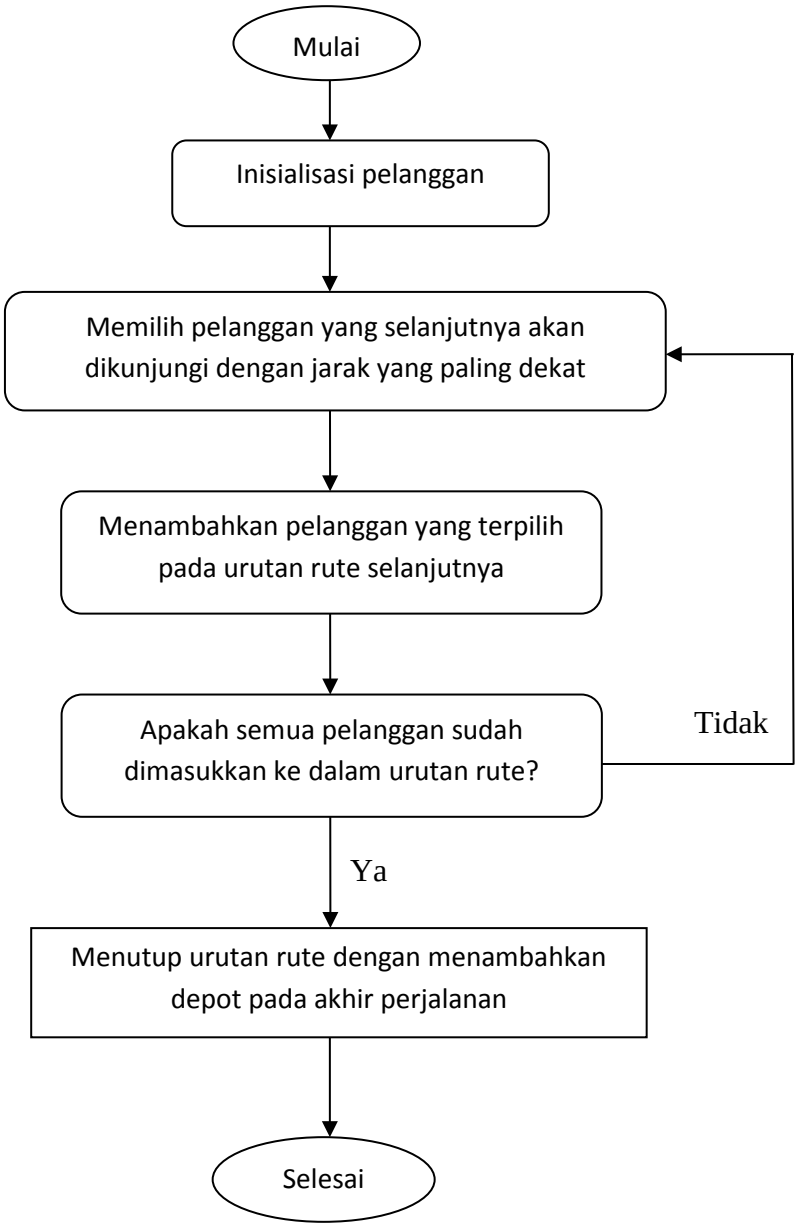
   Kepelangganlain yang memiliki jarak terdekat dari pelanggan yang terpilih sebelumnya dan jumlah pengiriman tidak melebihi kapasitas kendaraan.

   a. Apabila ada pelanggan yang terpilih sebagai pelanggan berikutnya dan terdapat sisa kapasitas kendaraan, kembali ke langkah (2).
   b. Bila kendaraan tidak memiliki sisa kapasitas, kembali ke langkah (1).
   c. Bila tidak ada lokasi yang terpilih karena jumlah pengiriman melebihi kapasitas kendaraan, maka kembali ke langkah (1). Dimulai lagi dari depot dan mengunjungi pelanggan yang belum dikunjungi yang memiliki jarak terdekat.

3. Langkah 3

   Bila semua pelanggan telah dikunjungi tepat satu kali maka algoritma berakhir.

Berikut diagram alir metode *Nearest Neighbour* :




Gambar2.16 Diagram Alir Metode *Nearest Neighbour*

Wahyu Kartika (2015)