OPTIMALISASI PROSES DISTRIBUSI BARANG BERDASARKAN ALGORITMA *TILLMAN & COCHRAN*

SKRIPSI

diajukan guna melengkapi tugas akhir dan memenuhi salah satu syarat
untuk menyelesaikan Program Sarjana Sains (S1)
dan mencapai gelar Sarjana Sains

Oleh

Bayu Arif Setiawan
NIM 001810101034

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2007

PERSEMBAHAN

Dengan menyebut nama Allah Yang Maha Pengasih dan Maha Penyayang serta sholawat kepada Nabi Muhammad SAW, dengan setulus hati kupersembahkan skripsi ini kepada :

1. Ayahanda Eddy Sujoko dan Ibunda Puji Lestari yang tak henti-hentinya memberi dorongan, nasehat, dan do'a hingga saat ini;
2. Kakek dan Nenek tercinta mbah Djasim, budhe Ninik dan seluruh keluarga besar di Banyuwangi, yang telah banyak memberikan dorongan baik moral maupun spiritual demi keberhasilan dan kebahagiaanku;
3. guru-guruku sejak SD sampai Perguruan Tinggi yang telah memberikan ilmu dan membimbing dengan penuh kesabaran;
4. Almamater Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

MOTTO

Sesungguhnya sesudah kesulitan itu ada kemudahan, maka apabila kamu telah selesai dari suatu urusan, kerjakanlah dengan sunguh-sungguh urusan yang lain. Dan hanya kepada Allah-lah hendaklah kamu berharap.
(QS Al-Insyirah : 6-7)

Renungkanlah apa yang telah kamu lakukan hari ini.
Jadikanlah ia sebagai pijakan untuk melangkah di esok hari.
(Author Unknown)

Janganlah takut mengalami sebuah kegagalan,
karena banyak hal yang dapat kita pelajari dari sebuah kegagalan.
(Author Unknown)

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

nama : Bayu Arif Setiawan

NIM : 001810101034

menyatakan dengan sesungguhnya bahwa karya ilmiah yang berjudul : *Optimalisasi Proses Distribusi Barang berdasarkan Algoritma Tillman & Cochran* adalah benar-benar hasil karya sendiri, kecuali jika dalam pengutipan substansi disebutkan sumbernya, dan belum pernah diajukan pada institusi manapun, serta bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa adanya tekanan dan paksaan dari pihak manapun serta bersedia mendapat sanksi akademik jika ternyata di kemudian hari pernyataan ini tidak benar.

Jember, 29 Oktober 2007

Yang menyatakan,

Bayu Arif Setiawan

NIM 001810101034

SKRIPSI

OPTIMALISASI PROSES DISTRIBUSI BARANG BERDASARKAN ALGORITMA *TILLMAN & COCHRAN*

Oleh

Bayu Arif Setiawan
NIM 001810101034

Pembimbing

Dosen Pembimbing Utama	: Firdaus Ubaidillah, S.Si, M.Si
Dosen Pembimbing Anggota	: Ahmad Kamsyakawuni, S.Si

PENGESAHAN

Skripsi berjudul *Optimalisasi Proses Distribusi Barang berdasarkan Algoritma Tillman & Cochran* telah diuji dan disahkan oleh Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember pada:

hari : SENIN
tanggal : 12 NOV 2007
tempat : Fakultas Matematika dan Ilmu Pengetahuan Alam

Tim Penguji

Ketua,

Firdaus Ubaidillah, S.Si, M.Si
NIP. 132 213 838

Sekretaris,

Ahmad Kamsyakawuni, S.Si
NIP. 132 206 038

Anggota I,

Drs. Moh. Hasan, M.Sc, Ph.D
NIP. 131 759 844

Anggota II,

Yuliani Setia Dewi, S.Si, M.Si
NIP. 132 258 183

Mengesahkan
Dekan FMIPA Universitas Jember

Ir. Sumadi, MS.
NIP. 130 368 784

RINGKASAN

Optimalisasi Proses Distribusi Barang berdasarkan Algoritma *Tillman & Cochran*; Bayu Arif Setiawan, 001810101034; 2007: 23 halaman; Jurusan Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Algoritma *Tillman & Cochran* adalah suatu metode untuk menyelesaikan masalah pendistribusian barang atau *Vehicle Routing Problem (VRP)*, yaitu mencari rute terpendek dalam pendistribusian barang sehingga dapat meminimumkan total jarak dan jumlah kendaraan pengangkut. Algoritma tersebut merupakan perluasan dari algoritma *Clarke & Wright.* Algoritma *Clarke & Wright* telah dibahas oleh Imagita (2006) dalam skripsinya dengan mengambil data dari PT. LIVIA MANDIRI SEJATI. Penelitian ini bertujuan untuk membandingkan kedua algoritma di atas, berdasarkan data yang telah digunakan sebelumnya. Dari data tersebut, hanya ada beberapa titik yang diketahui jaraknya dan dapat dikatakan bahwa titik-titik tersebut saling terhubung. Sedangkan untuk titik-titik yang lain, tidak diketahui apakah titik-titik tersebut saling terhubung. Untuk mengatasi masalah tersebut, diberikan asumsi bahwa titik-titik yang tidak diketahui jaraknya tersebut saling terhubung, dan ditetapkan jarak 2 buah titik yang tidak diketahui jaraknya sebesar jumlah jarak kedua titik tersebut dari titik asal.

Dari hasil penelitian dapat diketahui bagaimana rute pendistribusian barang dan jumlah armada yang diperlukan oleh PT LIVIA MANDIRI SEJATI, berdasarkan algoritma *Tillman & Cochran.* Kemudian dapat disimpulkan bahwa jumlah armada yang diperoleh dari algoritma tersebut lebih kecil daripada jumlah armada yang diperoleh dari algoritma *Clarke & Wright.* Akan tetapi total jarak yang diperoleh dari algoritma *Tillman & Cochran* jauh lebih besar daripada total jarak yang diperoleh dari algoritma *Clarke & Wright.*

PRAKATA

Puji syukur penulis panjatkan kehadirat Allah SWT atas rahmat, barokah, hidayah dan inayah-Nya, sehingga skripsi ini dapat penulis selesaikan. Penulisan skripsi ini dimaksudkan untuk memenuhi salah satu syarat guna memperoleh gelar kesarjanaan Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Penulis menyampaikan terima kasih dan penghargaan yang setinggi-tingginya kepada semua pihak yang telah banyak membantu dalam penyelesaian skripsi ini, diantaranya adalah :

1. Firdaus Ubaidillah, S.Si, M.Si selaku Dosen Pembimbing Utama dan Ahmad Kamsyakawuni, S.Si selaku Dosen Pembimbing Anggota yang bersedia meluangkan waktu untuk memberikan arahan serta masukan;
2. Drs. Moh. Hasan, M.Sc, Ph.D dan Yuliani Setia Dewi, S.Si, M.Si selaku Dosen Penguji yang telah memberikan kritik, saran dan masukan sehingga skripsi ini dapat terselesaikan dengan baik;
3. seluruh Dosen dan Civitas Akademika Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember;
4. kedua Orang Tuaku, Adikku, Putraku, dan semua keluargaku yang selalu memberikan motivasi serta dorongan moril, materiil dan spiritual;
5. kakakku Wirid yang telah banyak membantu dalam penyelesaian skripsi ini;
6. sahabat seperjuanganku (Martin & Moyan) yang terus-menerus memberikan motivasi dan selalu ada dalam suka maupun duka;
7. sahabat-sahabatku angkatan 2000 yang selalu memberikan motivasi, kritik maupun saran.

Penulis menyadari sepenuhnya bahwa dalam penyusunan skripsi ini masih jauh dari sempurna, oleh karena itu kritik dan saran yang membangun dari semua pihak sangat diharapkan. Akhirnya penulis berharap semoga skripsi ini bermanfaat bagi perkembangan ilmu pengetahuan, khususnya bidang Matematika.

Jember, Oktober 2007 Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Ilmu pengetahuan dan teknologi saat ini berkembang dengan sangat pesat terutama setelah ditemukannya komputer. Dengan membuat atau mendesain sebuah program penyelesaian berdasarkan algoritma atau langkah-langkah tertentu, banyak permasalahan dapat diselesaikan dengan komputer. Beberapa diantaranya adalah penyelesaian sebuah perhitungan matematika, pembuatan sebuah grafik atau kurva dari sebuah program penyelesaian, penyusunan jadual, misal jadual kuliah, jadual keberangkatan kendaraan umum, dan masih banyak lagi yang lainnya.

Salah satu permasalahan dalam matematika yang juga berhubungan dengan komputer adalah masalah rute kendaraan atau *Vehicle Routing Problem* (*VRP*). *VRP* merupakan salah satu kajian dalam bidang matematika yang membahas tentang bagaimana memilih beberapa rute yang harus dilalui oleh sejumlah kendaraan pengangkut dalam proses pendistribusian barang, yang mengkombinasikan permintaan tiap *retailer*, dengan memperhatikan kapasitas angkut kendaraan. *VRP* dapat diselesaikan dengan algoritma *Clarke & Wright* yang dikembangkan oleh *Clarke & Wright* pada tahun 1964 (Doerner, 2001). Prinsip dasar dari algoritma tersebut adalah bagaimana menentukan pasangan-pasangan *retailer* yang mempunyai penghematan terbesar, dengan memilih salah satu dari beberapa kombinasi pasangan retailer. Penghematan tersebut bisa berupa jarak, waktu, maupun biaya. Algoritma *Clarke & Wright* telah dibahas oleh Imagita (2006) dalam skripsinya dengan mengambil data dari PT. LIVIA MANDIRI SEJATI. Dalam skripsi tersebut dijelaskan bagaimana menentukan rute pendistribusian barang dengan algoritma di atas, dan bagaimana menentukan jadwal pengiriman barang dengan aturan *Large Processing Time*, dengan asumsi bahwa depo melayani lebih dari satu jenis produk dengan kapasitas angkut kendaraan adalah sama.

Selain algoritma *Clarke & Wright*, ada beberapa algoritma yang juga dapat digunakan untuk menyelesaikan masalah *VRP*, salah satu diantaranya adalah

algoritma *Tillman & Cochran*. Algoritma ini merupakan perluasan dari algoritma *Clarke & Wright* yang diharapkan dapat digunakan untuk menjadualkan rute distribusi di berbagai bidang produksi. Kedua algoritma tersebut pada dasarnya sama-sama mempunyai nilai yang optimal. Perbedaan penting dari kedua algoritma tersebut adalah bahwa algoritma *Tillman & Cochran* memperbolehkan adanya pembatasan dalam sistem, dan untuk beberapa kasus, algoritma ini menghasilkan jawaban yang lebih baik. Sedangkan pada algoritma *Clarke & Wright*, tidak memperbolehkan adanya pembatasan dalam sistem (Tillman & Cochran, 1968).

1.2 Perumusan Masalah

Dari gambaran yang sudah tertulis di atas, muncul beberapa permasalahan yaitu :

1. bagaimana menyelesaikan masalah *VRP* berdasarkan algoritma *Tillman & Cochran*;
2. bagaimana mengaplikasikan algoritma *Tillman & Cochran* ke dalam sebuah program komputer;
3. bagaimana hasil yang diperoleh dari algoritma *Tillman & Cochran* jika dibandingkan dengan hasil yang diperoleh dari algoritma *Clarke & Wright* untuk data yang sama.

1.3 Batasan Masalah

Sebagaimana telah disebutkan di atas bahwa algoritma *Tillman & Cochran* memperbolehkan adanya batasan dalam sistem, untuk itu, dalam penelitian ini diberikan batasan berupa pengabaian kondisi jalan seperti kepadatan lalu lintas, *traffic light*, dan jalan yang rusak, serta diasumsikan bahwa kapasitas kendaraan pengangkut dan jenis muatan adalah sama.

1.4 Tujuan

Tujuan dari penelitian ini adalah untuk mendapatkan sebuah solusi masalah pendistribusian barang berdasarkan algoritma *Tillman & Cochran*, dan

mengaplikasikan algoritma tersebut ke dalam sebuah program komputer untuk mempermudah pencarian, kemudian membandingkan hasil yang diperoleh dari algoritma tersebut dengan hasil yang diperoleh dari algoritma *Clarke & Wright*.

1.5 Manfaat

Dengan adanya penelitian ini ada beberapa manfaat yang bisa diperoleh, yaitu :

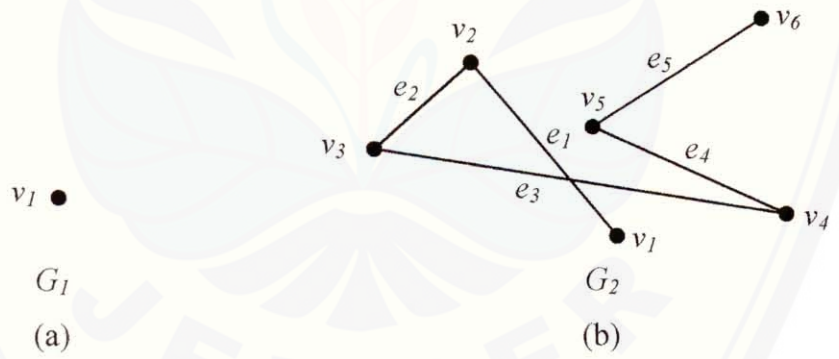
1. dapat diketahui solusi yang diperoleh dari algoritma *Tillman & Cochran* berupa rute pendistribusian barang, total jarak dan jumlah armada yang dibutuhkan dalam proses pendistribusian barang;
2. dapat diketahui perbandingan antara algoritma tersebut dengan algoritma *Clarke & Wright* yang dapat digunakan sebagai acuan bagi perusahaan dalam menentukan rute distribusi barang.

BAB 2. TINJAUAN PUSTAKA

2.1 Terminologi Dasar Graf

Berikut ini diberikan definisi dan notasi dalam teori graf yang berkaitan dengan skripsi ini.

Sebuah graf G didefinisikan sebagai pasangan himpunan (V(G), E(G)) dengan $V(G) = \{v_1, v_2, v_3, \ldots, v_n\}$ adalah sebuah himpunan hingga tak kosong yang elemen-elemennya disebut titik (*vertex*), dan $E(G) = \{e_1, e_2, e_3, \ldots, e_n\}$ adalah himpunan hingga boleh kosong yang elemen-elemennya menunjukkan pasangan tak terurut dari dua elemen di V yang disebut sisi (*edge*), yaitu $e = \{u,v\}$ dengan $u,v \in V$ (Fletcher, 1991). Berdasarkan definisi ini, sebuah graf tidak harus mempunyai sisi tetapi harus mempunyai minimal satu titik. Sebagai contoh, Gambar 2.1 (a) merupakan graf G_1 dengan satu titik, dan Gambar 2.1 (b) merupakan graf G_2 dengan enam titik dan lima sisi yaitu $V(G_2) = \{v_1, v_2, v_3, v_4, v_5, v_6\}$ dan $E(G_2) = \{e_1, e_2, e_3, e_4, e_5\}$.

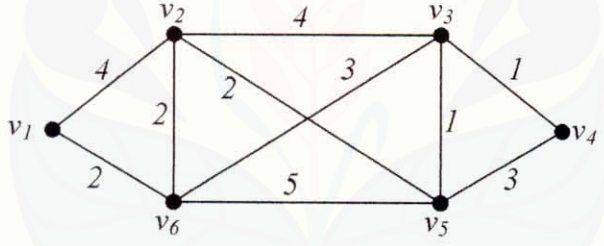


Gambar 2.1 (a) Graf dengan satu titik dan (b) Graf dengan enam titik dan lima sisi

Order dari graf G adalah banyaknya titik yang ada di G yaitu $n = |V(G)|$, dan graf yang mempunyai order berhingga disebut graf hingga. Sebagai contoh, graf G_2 pada Gambar 2.1 (b) adalah graf berorder enam. *Size* adalah banyaknya sisi yang ada pada graf G dan dinotasikan dengan $|E(G)|$. Derajat (*degree*) adalah banyaknya sisi yang menempel pada masing-masing titik v pada graf G dan dinotasikan dengan $deg(v)$.

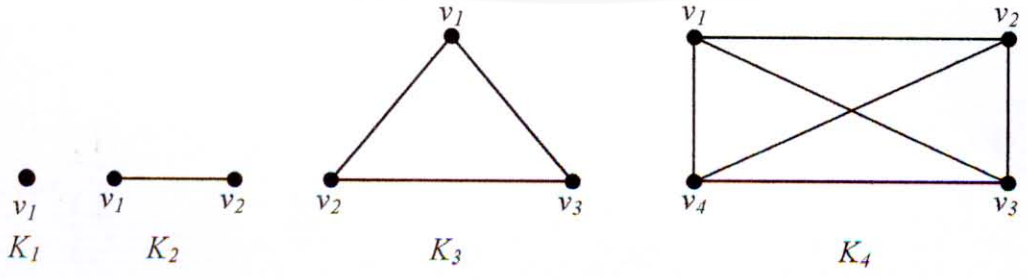
Sebuah jalan (*walk*) W dari v_0 ke v_n pada graf G dengan $W = v_1, e_1, v_2, e_2, v_3, e_3, v_4, e_4, \ldots\ldots, v_{i-1}, e_n, v_i$ adalah barisan berhingga (tak kosong) yang suku-sukunya bergantian antara titik dan sisi, sedemikian hingga v_{i-1} dan v_i adalah titik-titik akhir sisi e_n untuk $1 \leq i \leq n$ (Budiyasa, 1994). Suatu jalan yang barisan titik-titiknya tidak mengalami pengulangan disebut lintasan (*path*). Suatu jalan tertutup tanpa pengulangan titik kecuali titik awal dan titik akhir disebut sikel (*cycle*) (Chartrand & Oellermann, 1993), dengan kata lain, sikel adalah suatu lintasan yang tertutup. Suatu graf dikatakan terhubung jika setiap titik yang berlainan dihubungkan dengan sekurang-kurangnya sebuah lintasan. Sebuah lintasan dikatakan *Hamilton* jika lintasan tersebut melewati tepat satu kali semua titik yang ada pada graf tersebut, dan apabila lintasan tersebut adalah lintasan tertutup maka disebut *sikel Hamilton.*

Bobot dari suatu graf G adalah jumlah bobot dari semua sisi pada graf tersebut dan dinotasikan dengan $w(G)$. Berikut ini adalah contoh graf berbobot.



Gambar 2.2 Graf berbobot

Graf sederhana adalah graf yang tidak memiliki sisi ganda atau *loop*. Graf sederhana yang setiap titiknya mempunyai sisi ke semua titik yang lain disebut graf lengkap (Munir, 2001). Graf lengkap dengan n titik dinotasikan dengan K_n.



Gambar 2.3 Graf lengkap K_n, n = 1, 2, 3, 4

2.2 *Vehicle Routing Problem* (Masalah Rute Kendaraan)

Vehicle Routing Problem (VRP) atau masalah rute kendaraan merupakan salah satu kajian dalam ilmu matematika, khususnya dalam teori riset operasi yang banyak manfaatnya dalam aplikasi di dunia nyata, misalnya proses pendistribusian barang oleh sebuah perusahaan, proses pengumpulan surat dari kotak surat atau pengumpulan koin dari telepon umum, proses penentuan rute yang akan ditempuh oleh seorang salesman ataupun seorang dokter keliling, dan masih banyak lagi yang lainnya.

Secara umum, VRP dapat digambarkan sebagai berikut. Misalkan suatu depo mempunyai n-kendaraan dengan kapasitas angkut yang sama dan melayani sejumlah *retailer*. Depo tersebut ingin membuat m-rute yang akan dilalui oleh setiap kendaraan untuk melakukan pengiriman kepada sejumlah *retailer*. Perjalanan setiap kendaraan berawal dan berakhir pada depo. Permasalahannya adalah memilih beberapa rute dengan mengkombinasikan permintaan tiap *retailer* serta memperhatikan kapasitas angkut kendaraan. Dari rute yang terpilih ditentukan urutan *retailer* yang akan dikunjungi untuk meminimumkan total jarak maupun total waktu. Permasalahan inilah yang dikenal dengan istilah *vehicle routing problem* (*VRP*).

Dalam *VRP*, terdapat tiga unsur utama yaitu sebuah depo, sejumlah *retailer* yang akan dikunjungi dan sejumlah kendaraan yang dimiliki depo (Doerner, 2001). *VRP* didefinisikan ke dalam sebuah graf terhubung $G=(V,E)$ dimana $V=\{v_0,v_1,v_2,\ldots,v_n\}$ adalah himpunan titik dan $E=\{e_1,e_2,e_3,\ldots,e_n\}$ adalah himpunan sisi. Dalam hal ini v_0 melambangkan depo, v_i melambangkan *retailer* i, $e=(v_i,v_j)$ melambangkan jalan yang menghubungkan *retailer* i ke *retailer* j dengan bobot $w(v_i,v_j)$ adalah bilangan real positif yang menyatakan waktu tempuh atau jarak dari *retailer* i ke *retailer* j dan bobot $w(v_0,v_i)$ menyatakan waktu tempuh atau jarak dari depo ke *retailer* i. Diasumsikan bahwa graf G adalah graf sederhana dan tak berarah sehingga $w(v_i,v_j)= w(v_j,v_i)$ dan graf G juga merupakan graf *Hamilton*. Masalah rute kendaraan adalah bagaimana membentuk subgraf-subgraf yang merupakan graf terhubung dan graf *Hamilton* dari graf G, kemudian

bagaimana menentukan sikel *Hamilton* dengan bobot minimum dari setiap subgraf tersebut.

2.3 *Saving Methods* (Metode Penghematan *Clarke & Wright*)

Clarke & Wright mengusulkan sebuah metode sederhana untuk mengoptimumkan rute yang akan dilalui suatu armada truk, dengan kapasitas truk yang berbeda-beda, dari sebuah depot ke sejumlah pelanggan. Adapun langkah-langkah dalam metode ini antara lain adalah :

1. hubungkan masing-masing rute dari depot ke sebuah pelanggan;
2. kombinasikan dua buah pelanggan dalam sebuah rute jika jumlah permintaan tidak melebihi kapasitas angkut kendaraan;
3. hitung penghematan biaya, dan pilih penghematan yang terbesar;
4. jika kombinasi pada point 2 tidak dapat mengurangi total jarak, maka hapus dua buah rute dari dua buah pelanggan dan gantikan dengan dua buah rute yang baru;
5. hitung penghematan jarak yang berkaitan dengan perubahan rute tersebut;
6. pilih penghematan terbesar, jika tidak melebihi batas waktu atau kapasitas.

Dalam metode ini, penghematan dari kombinasi antara dua buah pelanggan i dan j dengan sebuah rute adalah :

$$S_{i,j} = d_{0i} + d_{0j} - d_{ij}$$

dimana d_{ij} menyatakan jarak dari pelanggan i ke pelanggan j, sedangkan pelanggan 0 menyatakan depo tersebut (Clarke & Wright, 1964).

2.4 Algoritma *Tillman & Cochran*

Algoritma diartikan sebagai urutan langkah-langkah logis penyelesaian masalah yang disusun secara matematis (Munir, 2001). Algoritma *Tillman & Cochran* merupakan suatu metode pencarian yang digunakan untuk menyelesaikan masalah rute kendaraan atau *Vehicle Routing Problem* (*VRP*). Algoritma ini merupakan perluasan dari algoritma *Clarke & Wright* yang diharapkan dapat digunakan untuk menjadualkan rute distribusi di berbagai bidang produksi. Kedua algoritma tersebut pada dasarnya sama-sama mempunyai

nilai yang optimal. Perbedaan penting dari kedua algoritma tersebut adalah bahwa algoritma *Tillman & Cochran* memperbolehkan adanya pembatasan dalam sistem, dan untuk beberapa kasus, algoritma ini menghasilkan jawaban yang lebih baik. Sedangkan pada algoritma *Clarke & Wright*, tidak memperbolehkan adanya pembatasan dalam sistem (Tillman & Cochran, 1968). Berikut adalah langkah-langkah algoritma *Tillman & Cochran* :

1. mengidentifikasi titik-titik tujuan, yaitu memberi nomor pada tiap-tiap titik tujuan, misalnya t_1, t_2, t_3, …, t_n;
2. mengidentifikasi order dari masing-masing titik tujuan;
3. mengidentifikasi jarak titik asal ke masing-masing titik tujuan;
4. cari jarak yang terkecil pada langkah tiga;
5. masukkan order atau pesanan titik tersebut;
6. cek muatan dan kapasitas
   a. jika jumlah muatan melebihi kapasitas maka order tersebut tidak dapat dikirim;
   b. jika jumlah muatan tidak melebihi kapasitas, maka kembali ke langkah tiga dengan mengasumsikan titik tersebut sebagai titik asal;
   c. jika jumlah muatan sama dengan kapasitas, maka kembali ke langkah tiga untuk mencari rute yang akan dilewati oleh truk kedua, ketiga, dan seterusnya;

(Tillman & Cochran, 1968).

2.5 Pemrograman dengan software *Borland Delphi 6.0*

Program komputer adalah suatu urutan instruksi untuk dilaksanakan komputer agar hasil tertentu dapat diperoleh. Jadi program komputer merupakan perwujudan atau implementasi dari algoritma yang ditulis dalam bahasa pemrograman tertentu sehingga dapat dilaksanakan oleh komputer (Munir, 1999).

Borland Delphi 6.0 merupakan bahasa pemrograman yang berorientasi objek. Objek dalam *Delphi* adalah tipe data yang meng-*enkapsulasi* (mengkombinasikan data dan fungsionalitas ke dalam sebuah unit tunggal) data dan operasi di dalam sebuah unit tunggal. *Pemrograman Berorientasi Objek*

merupakan perluasan dari pemrograman terstruktur yang mengutamakan pemakaian ulang program dan *enkapsulasi* data berdasarkan fungsinya (Martina, 2000).

Dasar bahasa pemrograman yang digunakan dalam *Delphi* adalah *Pascal*, sebuah bahasa yang didesain khusus oleh Niklaus Wirth untuk mengajarkan pemrograman terstruktur. Bahasa *Pascal* adalah bahasa yang *strongly-typed*, artinya variabel harus selalu diberi nilai yang tipenya sama dengan deklarasinya. Bila tidak, maka *kompiler* akan memberikan pesan kesalahan (Matcho, 1997).

2.5.1. Kontrol Program

Delphi mempunyai kontrol program yang digunakan untuk mengatur jalannya program satu per satu, karena dalam satu program terdapat beberapa pernyataan yang dipisahkan dengan tanda titik koma. Berikut ini adalah beberapa kontrol program dalam *Delphi* yang akan digunakan dalam skripsi ini :

a. pengulangan;

Kontrol program pengulangan ini ada tiga macam yaitu:

1) pengulangan **`While ... Do`**

Bentuk penulisannya adalah :

```
While <kondisi> Do
<Blok pernyataan>;
```

Struktur pengulangan ini digunakan untuk melakukan perulangan satu pernyataan atau satu blok pernyataan, selama kondisi bernilai benar;

2) pengulangan **`Repeat ... Until`**

Bentuk penulisannya adalah :

```
Repeat
<Blok pernyataan>;
until <Kondisi>
```

Struktur pengulangan ini menekankan pada kondisi berhenti, artinya pernyataan akan dilaksanakan berulangkali dan hanya akan berhenti jika kondisi terpenuhi (bernilai benar);

3) pengulangan **For...do**

Struktur ini digunakan untuk menghasilkan perulangan sejumlah kali tanpa penggunaan kondisi apapun. Bentuk umumnya ada dua macam yaitu: menaik (*ascending*) dan menurun (*descending*).

Bentuk penulisannya untuk menaik adalah:

for nilai awal **to** nilai akhir **do** <pernyataan>;

sedangkan yang menurun adalah sebagai berikut

for nilai akhir **downto** nilai awal **do** <pernyataan>;

b. pencabangan Bersyarat **If ... Then ... Else**

Pencabangan bersyarat ini digunakan untuk mencabang ke pilihan tertentu berdasar pengujian suatu nilai logika. Bentuk penulisannya seperti berikut ini,

If <kondisi> **Then** <pernyataan>;

atau

If <kondisi> **Then** <pernyataan.1> **Else** <Pernyataan.2>;

Pernyataan **If** yang pertama akan menguji <kondisi> disebelah kanannya, jika <kondisi> bernilai benar maka <pernyataan> akan dilaksanakan, jika salah maka akan melanjutkan ke pernyataan berikutnya. Bentuk **If** yang kedua akan menguji <kondisi> di sebelah kanannya, jika <kondisi> bernilai benar maka <pernyataan.1> akan dilaksanakan, jika <kondisi> bernilai salah maka <pernyataan.2> akan dilaksanakan.

2.5.2. Fungsi dan Prosedur

Fungsi (*function*) dan prosedur (*procedure*) adalah suatu sub program (*routin*) yang biasanya dipakai sebagai alat untuk melakukan tugas tertentu dan atau mendapatkan nilai tertentu. Fungsi sebenarnya hampir sama dengan prosedur, yang membedakan adalah fungsi mempunyai nilai kembalian (*return value*), sedangkan prosedur tidak mempunyai nilai kembalian. Prosedur ditulis sebagai suatu penyataan yang berdiri sendiri, berikut ini adalah contoh penulisan prosedur dalam program :

```
Procedure TForm1. FormCreate(Sender: TObject);
```

```
Var
   <variabel yang digunakan>;
Begin
   <blok pernyataan>;
End;
```

Berikut ini adalah beberapa fungsi dalam *Delphi* yang akan digunakan dalam skripsi ini :

a. **StrToInt**

Fungsi ini digunakan untuk mendapatkan nilai bertipe bilangan bulat (*integer*) dari suatu data bertipe *string*. Bentuk penulisannya seperti berikut :

```
StrToInt (S:String);
```

S adalah data tipe *string* yang akan diambil nilai *integer*-nya;

b. **IntToStr**

Fungsi ini digunakan untuk mendapatkan nilai bertipe *string* dari suatu data bertipe *integer*. Bentuk penulisannya seperti berikut :

```
IntToStr (S:integer);
```

S adalah data tipe *integer* yang akan diambil nilai *string*-nya.

BAB 4. KESIMPULAN DAN SARAN

4.1 Kesimpulan

Dari hasil di atas dapat diketahui bahwa dengan menggunakan algoritma *Tillman & Cochran* untuk data yang sama dapat memperkecil jumlah armada. Akan tetapi total jarak dengan algoritma tersebut jauh lebih besar daripada total jarak dengan algoritma *Clarke & Wright*.

4.2 Saran

Asumsi yang digunakan pada skripsi ini adalah depo melayani permintaan *retailer* untuk satu jenis produk dengan kapasitas angkut semua kendaraan adalah sama. Algoritma *Tillman & Cochran* dapat dikembangkan dengan asumsi yang berbeda, misalnya masing-masing titik tidak semuanya saling terhubung, depo melayani *retailer* lebih dari satu jenis produk dengan kapasitas angkut setiap kendaraannya berbeda.

DAFTAR PUSTAKA

Budiyasa, I. (1994). ***Matematika Diskrit I,*** Institut Keguruan dan Ilmu Pendidikan, Surabaya.

Chartrand, G. and O.R. Oelermann. (1993). ***Applied and Algoritmic Graph Theory***. McGraw_Hill, Inc. New York.

Clarke, G. and Wright, J.W. (1964). ***Scheduling of Vehicles from A Central Depot to A Number of Delivery Points***, Operations Research, Vol. 12, p. 568-581.

Doerner, K. (2001). ***Savings Ants for the Vehicle Routing Problem***, Report No. 63, December.

Fletcher, P,. H. Hoyle, and C.W. Patty. (1991). ***Foundation of Discrete Mathematics***. PWS-KENT Publising Company, Boston.

Imagita, E. (2006). ***Optimasi Jarak Dan Total Waktu Pendistribusian Barang Menggunakan Algoritma Clarke-Wright Dan Aturan Prioritas Large Processing Time***, Lab. Komputasi Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember, Jember.

Martina, I. (2000). ***36 Jam Belajar Komputer Delphi 5.0.*** PT Elex Media Komputindo, Jakarta.

Matcho, J. and Faulkner, D. R. (1997). ***Panduan Penggunaan Delphi***, Andi, Yogyakarta.

Munir, R. (1999). ***Algoritma dan Pemrograman Bahasa Pascal dan C***, CV. Informatika, Bandung.

Munir, R. (2001). ***Matematika Diskrit***. CV. Informatika. Bandung.

Tillman, F.A. and Cochran, H. (1968). ***A Heuristic Approach for solving the Delivery Problem***, The Journal of Industrial Engineering, Vol. 19, No. 7, p. 354-358.

Lampiran A, Data Jarak antar titik

Kode Titik	Jarak antar Titik (km)
0 - 1	9,3
0 - 2	9,2
0 - 3	9,1
0 - 4	8,92
0 - 5	8,9
0 - 6	10,2
0 - 7	10
0 - 8	8,5
0 - 9	8,2
0 - 10	8,3
0 - 11	8,1
0 - 12	8,4
0 - 13	8,3
0 - 14	8,31
0 - 15	8,5
0 - 16	9,5
0 - 17	7,2
0 - 18	7,3
0 - 19	0,52
0 - 20	3
0 - 21	3,5
0 - 22	3,6
0 - 23	3,7
0 - 24	3,3
0 - 25	3,2
0 - 26	9
0 - 27	8,45
0 - 28	7,7
0 - 29	2
0 - 30	7,82
0 - 31	1,2
0 - 32	1,3
0 - 33	1
0 - 34	1
0 - 35	4
0 - 36	7,9
0 - 37	3,4
0 - 38	3,52
0 - 39	5,8
0 - 40	6
0 - 41	6,6
0 - 42	6,62

Kode Titik	Jarak antar Titik (km)
0 - 43	6,65
0 - 44	6,7
0 - 45	6,73
0 - 46	6,74
0 - 47	6,8
0 - 48	6,9
0 - 49	7,1
0 - 50	7,15
0 - 51	6,85
0 - 52	6,83
0 - 53	6,6
0 - 54	6,7
0 - 55	6,8
0 - 56	6,9
0 - 57	6,92
0 - 58	6,93
0 - 59	6,97
0 - 60	6,2
0 - 61	7,3
0 - 62	7,2
0 - 63	7,1
0 - 64	7,18
0 - 65	4,8
0 - 66	5,3
0 - 67	10,7
0 - 68	11
0 - 69	10,5
0 - 70	10,6
0 - 71	10,7
0 - 72	11
0 - 73	10,71
0 - 74	10,8
0 - 75	11,1
0 - 76	11
0 - 77	10,3
0 - 78	10,5
0 - 79	10,6
0 - 80	11,2
0 - 81	11,8
0 - 82	12
0 - 83	12
0 - 84	7,7

Kode Titik	Jarak antar Titik (km)
0 - 85	7,4
0 - 86	7,8
0 - 87	7,9
0 - 88	7,2
0 - 89	5,7
0 - 90	5,8
0 - 91	6
0 - 92	6,2
0 - 93	6,4
0 - 94	6,42
0 - 95	6,5
0 - 96	5,4
0 - 97	5,8
0 - 98	5,84
0 - 99	5,4
0 - 100	5,5
0 - 101	6
0 - 102	6,1
0 - 103	6,2
0 - 104	6,6
0 - 105	6,62
0 - 106	6,65
0 - 107	6,68
0 - 108	6,72
0 - 109	6,95
0 - 110	7,2
0 - 111	7,26
0 - 112	7,4
0 - 113	7,38
0 - 114	6,7
0 - 115	6,55
0 - 116	6,7
0 - 117	6,8
0 - 118	6,62
0 - 119	4
0 - 120	4,2
0 - 121	5,93
0 - 122	4
0 - 123	4,3
0 - 124	4,4
0 - 125	5,2
0 - 126	5,7

Kode Titik	Jarak antar Titik (km)
0 - 127	5,6
0 - 128	5,62
0 - 129	5,58
0 - 130	5,57
0 - 131	5,53
0 - 132	5,51
0 - 133	5,9
0 - 134	4,95
0 - 135	4,92
0 - 136	6,4
0 - 137	6,45
0 - 138	6,53
0 - 139	5,43
0 - 140	6,2
0 - 141	7,2
0 - 142	7,1
0 - 143	7,21
0 - 144	6,9
0 - 145	4
0 - 146	4,4
0 - 147	4,5
0 - 148	4,9
0 - 149	0,72
0 - 150	5,25
0 - 151	5,9
0 - 152	6,1
0 - 153	4,81
0 - 154	4,6
0 - 155	5,7
0 - 156	4,7
0 - 157	8
0 - 158	8,3
0 - 159	6,6
0 - 160	7,2
0 - 161	6,9
0 - 162	7,3
0 - 163	6,95
0 - 164	0,5
0 - 165	0,8
0 - 166	6,5
0 - 167	8,2
0 - 168	8,3
0 - 169	8,32
0 - 170	10,6

Kode Titik	Jarak antar Titik (km)
0 - 171	0,7
0 - 172	11,4
0 - 173	10,1
0 - 174	9
0 - 175	8,8
0 - 176	9,9
0 - 177	6,8
0 - 178	6,5
0 - 179	0,55
0 - 180	0,68
0 - 181	0,6
0 - 182	0,75
0 - 183	0,65
1 - 2	0,1
1 - 3	0,2
1 - 16	0,2
1 - 26	0,3
2 - 3	0,1
2 - 5	0,3
2 - 6	1
2 - 8	0,7
3 - 8	0,6
3 - 16	0,4
4 - 5	0,02
4 - 6	1,28
4 - 7	1,02
4 - 8	0,42
5 - 6	1,3
5 - 7	1,1
5 - 8	0,4
6 - 7	0,2
8 - 9	0,4
8 - 11	0,2
9 - 10	0,2
9 - 11	0,1
9 - 27	0,35
10 - 11	0,1
11 - 12	0,1
12 - 13	0,1
12 - 27	0,15
13 - 14	0,09
13 - 15	0,2
13 - 27	0,05
14 - 18	1,01

Kode Titik	Jarak antar Titik (km)
14 - 27	0,14
14 - 30	0,49
15 - 27	0,05
15 - 30	0,68
16 - 26	0,1
17 - 18	0,2
17 - 19	6,68
17 - 28	0,5
17 - 35	3,2
18 - 36	0,6
19 - 20	2,48
19 - 21	2,98
19 - 29	1,48
19 - 33	0,48
19 - 34	0,52
19 - 35	3,48
20 - 25	0,2
20 - 29	1
20 - 31	2
21 - 22	0,1
21 - 25	0,3
21 - 35	0,6
21 - 37	0,1
22 - 23	0,1
22 - 38	0,08
23 - 37	0,3
23 - 38	0,18
24 - 25	0,1
24 - 37	0,1
28 - 35	3,7
29 - 31	1,02
29 - 33	1
29 - 34	1
30 - 36	0,08
31 - 32	0,1
31 - 33	0,2
31 - 34	0,2
32 - 33	0,3
39 - 40	0,2
39 - 57	1,2
40 - 41	0,6
40 - 44	0,7
41 - 42	0,02
41 - 44	0,1

Kode Titik	Jarak antar Titik (km)
41 - 53	0,25
41 - 55	0,4
42 - 43	0,03
42 - 45	0,11
43 - 46	0,09
43 - 51	0,2
43 - 52	0,18
44 - 45	0,03
45 - 46	0,01
46 - 47	0,06
46 - 52	0,26
47 - 51	0,08
47 - 52	0,19
48 - 51	0,08
48 - 49	0,2
49 - 50	0,05
49 - 51	0,25
50 - 51	0,3
50 - 52	0,32
51 - 52	0,09
53 - 55	0,2
53 - 54	0,1
53 - 56	0,2
54 - 55	0,2
54 - 56	0,2
54 - 58	0,23
55 - 56	0,1
55 - 57	0,12
56 - 57	0,06
56 - 58	0,22
58 - 59	0,04
58 - 60	0,93
59 - 60	0,77
60 - 63	0,1
60 - 64	0,18
61 - 62	0,1
61 - 64	0,23
61 - 66	2
62 - 63	0,1
62 - 65	2,4
62 - 66	1,5
63 - 64	0,18
65 - 66	0,5
67 - 68	0,3

Kode Titik	Jarak antar Titik (km)
68 - 69	0,2
69 - 77	0,2
69 - 83	1,5
69 - 85	3,3
70 - 71	0,1
70 - 73	0,2
70 - 77	0,4
71 - 73	0,25
72 - 74	0,2
72 - 75	0,45
73 - 74	0,2
73 - 78	0,21
74 - 75	0,3
74 - 78	0,3
75 - 79	0,52
76 - 79	0,4
76 - 80	0,2
76 - 81	0,8
77 - 78	0,2
77 - 85	3,1
78 - 79	0,1
79 - 80	0,6
80 - 81	0,6
81 - 82	0,2
82 - 83	0,1
84 - 87	0,3
84 - 88	0,5
85 - 86	0,4
85 - 87	0,8
85 - 88	0,2
86 - 87	0,5
89 - 90	0,1
89 - 91	0,3
89 - 95	0,8
89 - 98	0,7
89 - 99	0,3
89 - 100	0,5
89 - 101	0,3
90 - 91	0,2
90 - 92	0,4
90 - 101	0,2
91 - 93	0,4
91 - 95	0,5
91 - 114	0,3

Kode Titik	Jarak antar Titik (km)
91 - 115	0,25
92 - 118	0,42
93 - 94	0,05
93 - 114	0,3
93 - 115	0,15
93 - 118	0,22
94 - 95	0,08
96 - 100	0,26
96 - 101	0,6
96 - 102	0,7
96 - 117	1,4
96 - 121	0,53
97 - 103	0,9
97 - 121	0,13
98 - 100	0,34
98 - 120	1,64
98 - 121	0,09
99 - 100	0,1
99 - 101	0,6
99 - 120	1,2
100 - 120	1,3
101 - 102	0,1
102 - 103	0,1
103 - 104	0,4
103 - 117	0,6
104 - 105	0,02
104 - 117	0,2
105 - 106	0,02
105 - 117	0,18
106 - 107	0,03
107 - 109	0,27
107 - 108	0,21
107 - 116	0,2
108 - 109	0,23
108 - 110	0,48
108 - 116	0,02
109 - 110	0,25
109 - 113	0,43
110 - 111	0,06
110 - 113	0,18
111 - 112	0,14
111 - 113	0,12
112 - 113	0,05
114 - 115	0,5

Kode Titik	Jarak antar Titik (km)
115 - 118	0,07
116 - 117	0,1
119 - 120	0,2
122 - 123	0,3
122 - 124	0,4
122 - 145	0,34
122 - 149	3,28
123 - 124	0,1
123 - 133	1,6
124 - 125	0,8
124 - 132	1,11
125 - 126	0,5
125 - 132	0,31
125 - 133	0,7
125 - 136	1
126 - 127	0,1
126 - 132	0,19
127 - 128	0,2
127 - 130	0,07
127 - 131	0,09
128 - 129	0,06
129 - 130	0,1
130 - 131	0,05
131 - 132	0,03
133 - 136	1,2
134 - 135	0,27
134 - 147	0,45
134 - 151	0,95
157 - 167	0,2
157 - 169	0,2
157 - 176	1,9
158 - 168	1
158 - 169	1,2
158 - 177	1,5
159 - 163	0,35
159 - 166	0,43
159 - 177	0,2
159 - 178	0,4
160 - 161	0,35
160 - 162	0,1
160 - 163	0,3
160 - 178	0,5
161 - 162	0,3
161 - 163	0,05

Kode Titik	Jarak antar Titik (km)
162 - 163	0,35
162 - 165	5,7
162 - 178	0,6
163 - 166	0,45
164 - 179	0,05
164 - 181	0,1
165 - 166	6,4
165 - 182	0,05
166 - 168	1,8
166 - 171	5,8
167 - 168	0,1
167 - 169	0,21
167 - 177	1,4
168 - 169	0,1
168 - 177	1
169 - 175	1
170 - 172	1,3
170 - 173	0,5
170 - 176	0,7
171 - 180	0,06
171 - 182	0,05
171 - 183	0,05
172 - 173	0,8
172 - 174	2,4

Lampiran B. Data Permintaan

Kode Titik	Jumlah Order
1	7
2	7
3	6
4	8
5	5
6	9
7	7
8	6
9	8
10	9
11	5
12	6
13	8
14	7
15	7
16	7
17	8
18	14
19	9
20	6
21	7
22	7
23	6
24	8
25	8
26	14
27	15
28	14
29	8
30	9
31	7
32	9
33	5
34	7
35	6
36	7
37	7
38	6
39	11
40	11
41	14
42	8
43	10
44	12
45	12
46	10
47	10
48	12
49	9

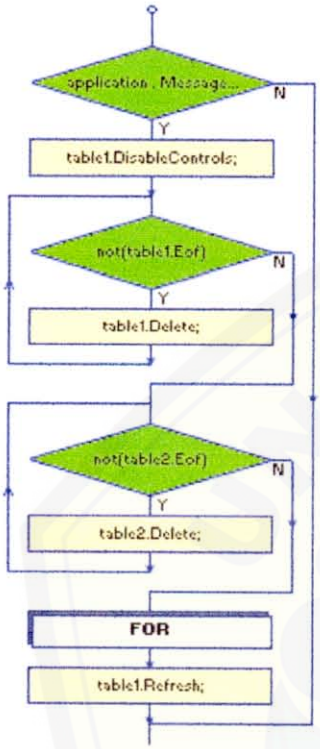
Kode Titik	Jumlah Order
50	13
51	13
52	9
53	8
54	14
55	11
56	11
57	11
58	14
59	8
60	10
61	12
62	11
63	9
64	13
65	7
66	7
67	17
68	13
69	15
70	11
71	15
72	13
73	14
74	12
75	11
76	14
77	13
78	12
79	15
80	15
81	11
82	13
83	13
84	12
85	14
86	13
87	18
88	16
89	9
90	10
91	8
92	7
93	11
94	9
95	7
96	11
97	8
98	10

Kode Titik	Jumlah Order
99	7
100	11
101	6
102	12
103	12
104	6
105	6
106	6
107	12
108	12
109	11
110	7
111	9
112	9
113	9
114	9
115	8
116	7
117	10
118	7
119	9
120	9
121	12
122	9
123	8
124	7
125	8
126	9
127	8
128	11
129	10
130	7
131	9
132	6
133	9
134	8
135	7
136	8
137	8
138	8
139	10
140	9
141	6
142	10
143	10
144	6
145	6
146	7
147	9

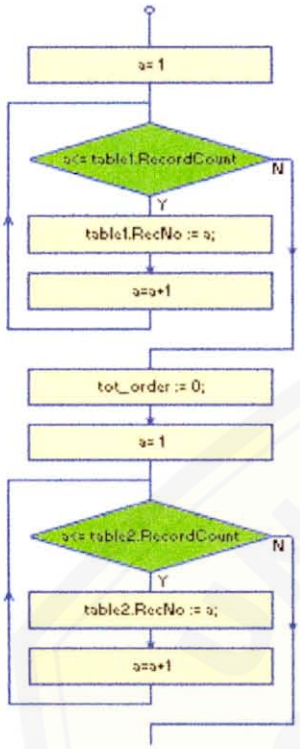
Kode Titik	Jumlah Order
148	10
149	8
150	10
151	8
152	9
153	9
154	8
155	7
156	13
157	11
158	15
159	13
160	14
161	12
162	12
163	15
164	5
165	6
166	13
167	13
168	13
169	14
170	12
171	6
172	11
173	14
174	12
175	13
176	12
177	14
178	13
179	5
180	9
181	7
182	9
183	5

Lampiran C. Flowchart

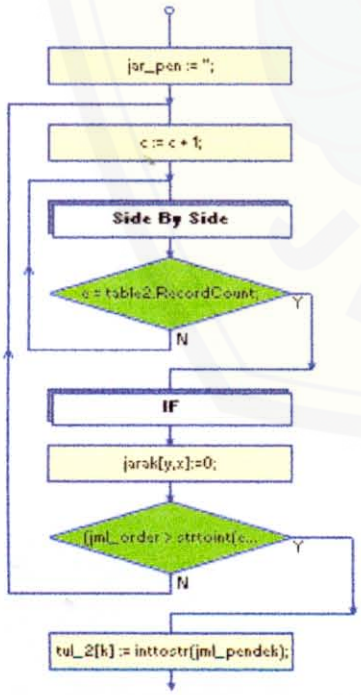
Flowchart dari Prosedur TVRASS_Form.SimpanClick



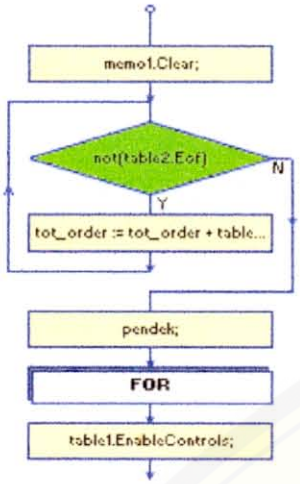
Flowchart dari Prosedur TVRASS_Form.Pendek



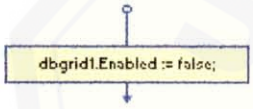
Flowchart dari Prosedur TVRASS_Form.Proses



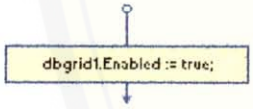
Flowchart dari Prosedur TVRASS_Form.AProsesClick



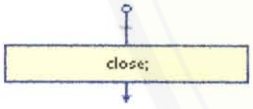
Flowchart dari Prosedur TVRASS_Form.DataBaruClick



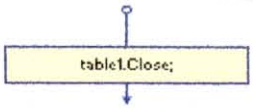
Flowchart dari Prosedur TVRASS_Form.BatalClick



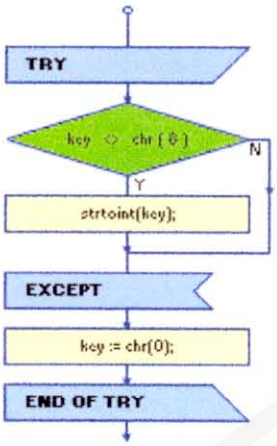
Flowchart dari Prosedur TVRASS_Form.TutupClick



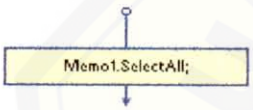
Flowchart dari Prosedur TVRASS_Form.FormClose



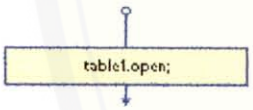
Flowchart dari Prosedur TVRASS_Form.Edit1KeyPressed



Flowchart dari Prosedur TVRASS_Form.Copy1Click



Flowchart dari Prosedur TVRASS_Form.FormCreate



Lampiran D. Script Program

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics,
Controls, Forms,
  Dialogs, StdCtrls, DB, DBTables, Grids, DBGrids, ExtCtrls,
Buttons, Menus;

  type
  TIntSet = set of 1..255;
  TVRASS_Form = class(TForm)
    DataSource1: TDataSource;
    DataSource2: TDataSource;
    DBGrid1: TDBGrid;
    DBGrid2: TDBGrid;
    Table1: TTable;
    Table2: TTable;
    AProses: TButton;
    Memo1: TMemo;
    Label4: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Panel1: TPanel;
    Label3: TLabel;
    Edit3: TEdit;
    Simpan: TButton;
    Data_Baru: TButton;
    Batal: TButton;
    Tutup: TBitBtn;
    Label1: TLabel;
    Label2: TLabel;
    Edit2: TEdit;
    Edit1: TEdit;
    PopupMenu1: TPopupMenu;
    Copy1: TMenuItem;
    procedure SimpanClick(Sender: TObject);
    procedure Pendek;
    procedure Proses;
    procedure AProsesClick(Sender: TObject);
    procedure Data_BaruClick(Sender: TObject);
    procedure BatalClick(Sender: TObject);
    procedure TutupClick(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action:
TCloseAction);
    procedure Edit1KeyPress(Sender: TObject; var Key: Char);
    procedure Copy1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
    jml_order, jml_pendek : integer;
```

```
    tot_Order, t_order  : integer;
    j_titik             : word;
    jarak               : array[0..1000,0..1000] of word;
    j_pendek            : array[1..1000] of word;
    tujuan              : array[1..1000] of word;
    order               : array[1..1000] of word;
    check               : array[1..1000] of word;
    tul_1, tul_2, tul_3 : array[1..1000] of string;
    x,y, k              : word;
  end;

var
  VRASS_Form: TVRASS_Form;

implementation

{$R *.dfm}

procedure TVRASS_Form.SimpanClick(Sender: TObject);
var
  a,b : word;
begin
  if application.MessageBox('Masukkan Data Baru?','Perubahan
Data',MB_YESNO or MB_ICONQUESTION)=IDYES Then
  Begin
  table1.DisableControls
  table2.DisableControls;
  table1.First;
  table2.First;

  while not(table1.Eof) do table1.Delete;
  while not(table2.Eof) do table2.Delete;

  for a := 0 to strtoint(edit3.Text) do
   Begin
    for b := 0 to strtoint(edit3.Text) do
    if a < b then
    Begin
      table1.Append;
      table1.FieldByName('T_1').Value := a;
      table1.FieldByName('T_2').Value := b;
      table1.Post;
    End;

    if a > 0 then
    Begin
      table2.Append;
      table2.FieldByName('Titik').value := a;
      Table2.Post;
    End;
   End;

  table1.Refresh;
  table2.Refresh;
  table1.First;
  table2.First;
```

```
  table1.EnableControls;
  table2.EnableControls;

  Batal.Click;
  End;
end;

procedure TVRASS_Form.Pendek;
var
  a : integer;
begin
  for a := 1 to table1.RecordCount do
  begin
    table1.RecNo := a;
    jarak[table1.Fields[0].AsInteger,table1.Fields[1].AsInteger]
:= table1.Fields[2].ASInteger;
  End;

  tot_order := 0;
  table2.open;
  table2.First;

  for a := 1 to table2.RecordCount do
  Begin
    table2.RecNo := a;
    order[table2.Fields[0].AsInteger] :=
table2.Fields[1].AsInteger;
    tot_Order := tot_order + order[a];

    check[a] := 0;
  End;
end;

procedure TVRASS_Form.Proses;
var
 a, b, c, d, e          : word;
 T_2                    : TIntSet;
  jar_pen, Jum_order    : string;
begin
  jar_pen := '';
  Jum_order := '0';

  k := k + 1;
  x := 0;  y := 1;
  a := x;
  jml_order := 0;
  jml_pendek := 0;
  c := 0;
  repeat
    c := c + 1;
    begin
    T_2 := [];
    e := 0;
    repeat
    j_Pendek[c] := 10000;
```

```
for b := 1 to strtoint(edit3.Text) do
if (check[b]<>1) and not(b in T_2) then
Begin
   if (a <> b) and(a > b) then
       d := jarak[b,a]
   else
        d := jarak[a,b];

   if (check[b]=0)  then
   if (j_Pendek[c] > d) and (d <> 0) then
      Begin
        j_Pendek[c] := d;
        x := a; y := b;
      End
   else
      Begin
        J_Pendek[c] := J_Pendek[c];
        x := x; y := y;
      End;
End;

if jml_order + order[y] > strtoint(edit2.text) then
   Begin
   T_2 := T_2 + [y];

   end;

e := e + 1;
until e = table2.RecordCount;

if jml_order + order[y] > strtoint(edit2.text) then
else
Begin
  jarak[x,y] := 0;  jarak[y,x]:=0;
  tujuan[c] := y;
  check[y] := 1;
  j_titik := j_titik + 1;

  jml_Order := jml_order + order[y];
  jml_pendek := jml_pendek + j_pendek[c];

  a := y;
  jar_pen := jar_pen +' + '+inttostr(J_Pendek[c]);
  jum_order := jum_order +'-->'+ inttostr(tujuan[c]);
End;

end;

until (jml_order > strtoint(edit2.Text)) or (c =
strtoint(edit3.Text))
        or (J_titik = strtoint(edit3.Text));

tul_2[k] := inttostr(jml_pendek);
tul_1[k] := jum_order;
tul_3[k] := inttostr(jml_order);
```

```
end;

procedure TVRASS_Form.AProsesClick(Sender: TObject);
var
  a, b : word;
begin
 memo1.Clear;
 VRASS_Form.Cursor :=  crSQLWait;
 table1.Open;
 table2.Open;
 table1.DisableControls;
 table2.DisableControls;
 table2.First;

 tot_order := 0;
 while not(table2.Eof) do
 begin
 tot_order := tot_order + table2.fieldbyname('order').AsInteger;
 table2.Next;
 end;

    Begin
      pendek;
      k := 0;
      t_order := 0;
      j_titik := 0;
      for a := 1 to table2.recordcount{strtoint(edit1.Text)} do
        Begin
          if t_order < Tot_order Then
          Begin
          Proses;
          t_order := t_order + jml_order;
          b := a mod strtoint(edit1.Text);
          if b = 0 then
             Begin
             memo1.Lines.Add('');
             memo1.Lines.Add('Armada '+
inttostr(b+strtoint(edit1.Text)));
             memo1.Lines.Add('-------------');
             memo1.Lines.Add('Rute    = '+ tul_1[k]);
             memo1.Lines.Add('Total jarak = '+ tul_2[k]);
             memo1.Lines.Add('Total order = '+ tul_3[k]);
             memo1.Lines.Add('-----------------------------------
----------------------------------------');
             memo1.Lines.Add('');
             end;
          else
             if k <> 1 then memo1.Lines.Add('');
             memo1.Lines.Add('Armada '+ inttostr(b));
             memo1.Lines.Add('-------------');
             memo1.Lines.Add('Rute    = '+ tul_1[k]);
             memo1.Lines.Add('Total jarak = '+ tul_2[k]);
             memo1.Lines.Add('Total order = '+ tul_3[k]);

          End;
        End;
```

```
    end;
    table1.EnableControls;
    table2.EnableControls;
    VRASS_Form.Cursor :=  crDefault;
end;

procedure TVRASS_Form.Data_BaruClick(Sender: TObject);
begin
  dbgrid1.Enabled := false;
  dbgrid2.Enabled := false;
  memo1.Enabled := false;

  data_baru.Enabled := false;
  AProses.Enabled := false;
  tutup.Enabled := false;

  table1.open;table2.open;
  Panel1.Visible := true;
  edit3.SetFocus;

  Edit3.Text := inttostr(table2.recordcount);
end;

procedure TVRASS_Form.BatalClick(Sender: TObject);
begin
  dbgrid1.Enabled := true;
  dbgrid2.Enabled := true;
  memo1.Enabled := true;

  Data_Baru.Enabled := true;
  AProses.Enabled := true;
  tutup.Enabled := true;

  panel1.Visible := false;
end;

procedure TVRASS_Form.TutupClick(Sender: TObject);
begin
  close;
end;

procedure TVRASS_Form.FormClose(Sender: TObject; var Action:
TCloseAction);
begin
  table1.Close;
  table2.Close;
end;

procedure TVRASS_Form.Edit1KeyPress(Sender: TObject; var Key:
Char);
begin
  try
    if key <> chr(8) then strtoint(key);
  except
    key := chr(0);
  end;
```

```
end;

procedure TVRASS_Form.Copy1Click(Sender: TObject);
begin
  Memo1.SelectAll;
  Memo1.CopyToClipboard;
end;

procedure TVRASS_Form.FormCreate(Sender: TObject);
begin
  table1.open;
  table2.open;
  edit3.Text := inttostr(table2.recordcount);
end;

end.
```