MENENTUKAN RUTE TERPENDEK DENGAN MENGGUNAKAN ALGORITMA FLOYD-WARSHALL

Gio Fandi H.Nainggolan
Fakultas Informatika
Universitas Nasional
e-mail : gionainggolan1@gmail.com

ABSTRAK

Masalah pendistribusian pada perusahaan adalah masalah yang sangat penting untuk diperhatikan. Pada dasarnya pendistribusian barang akan sangat menghemat perusahaan dalam berbagai hal.Pencarian rute terpendek yang dilakukan pada PT. Rapy Ray Putratama Medan dilakukan dengan menghubungkan berbagai macem outlet dan juga termasuk beberapa outletnya adalah PT. Rapy Ray Putratama cabang medan. Permasalah rute terpendek ini dapat disesaikan dengan menggunakan salah satu metode pencarian rute terpendek yaitu algoritma Floyd-Warshall. Penelitian ini bertujuan untuk mengetahui hasil dari rute yang akan dipilih sebagai saran atau masukan kepada Perusahaan. Untuk hasil pencarian rute terpendek dengan menggunakan algoritma Floyd-Warshallyang diimplementasikan dalam pemrograman Codeblocks:: adalah jarak dari PT ke outlet maupun dari outlet ke outlet memiliki jarak yang paling minimum. Setelah itu, dapat ditentukan rute terpendek yang akan dipilih oleh salesman dalam pendistribusian yang telah didapatkan pada program tersebut. Data yang diinput adalah data jarak. Output yang dihasilkan program adalah jarak terpendek. Dengan penghematan jarak yang telah dilakukan. Pembentukan rute usulan yang dihasilkan dengan menggunakan metode algoritma Floyd-Warshall menghasilkan rute yang lebih dekat dengan total jarak penghematan adalah 10.97 % (51.304 km).

Kata kunci*: Pendistribusian, Pencarian rute terpendek, algoritma Floyd-Warshall.*

1. Pendahuluan

Graf merupakan salah satu cabang ilmu matematika yang dapat digunakan dalam membantu persoalan diberbagai bidang seperti masalah komunikasi, transportasi, distribusi, aliran air, aliran listrik dan lain sebagainya. Salah satu kegunaan graf yang cukup penting adalah dalam hal pemilihan *path* terpendek dimana untuk mencari *path* terpendek dari simpul t (simpul awal) ke simpul s (simpul tujuan) adalah mencari jalur yang berbeda dari simpul t ke s dengan bobot yang seminimal mungkin. Bobot dalam graf adalah nilai yang diberikan pada setiap jalurnya. Bobot tersebut dapat menyatakan diameter, panjang, jarak antar tempat, waktu pengiriman, ongkos pengiriman dan lain sebagainya.

Transportasi atau pengangkutan adalah suatu kegiatan yang penting bagi kegiatan kita pada umumnya, dan pada kegiatan industri pada khususnya. Transportasi atau pengangkutan diartikan sebagai perpindahan barang dan manusia dari tempat asal ke tempat tujuan. Kasus transportasi timbul ketika kita mencoba menentukan cara pengiriman (pendistribusian) suatu jenis barang (item) dari beberapa sumber (lokasi penawaran) ke

beberapa tujuan (lokasi permintaan) yang dapat meminimumkan biaya [1].

Setiap perusahaan pasti menginginkan biaya yang minimum untuk proses transportasi ini sehingga diperlukan suatu strategi pemecahan masalah yang bisa memberikan solusi yang optimal. Selain jumlah barang dan biaya angkut, besarnya biaya pengiriman juga dipengaruhi oleh jarak yang ditempuh saat proses pendistribusian. Untuk itu diperlukan perencanaan yang matang agar jarak yang ditempuh sedikit dan produk akan diterima pelanggan dalam jumlah tepat dan dalam kondisi baik.

Sebenarnya ada banyak algoritma yang dapat menyelesaikan masalah pencarian rute terpendek, antara lain algoritma Dijkstra, algoritma Floyd-Warshall, dan algoritma Bellman-Ford. Akan tetapi merujuk pada jurnal/penelitian yang dilakukan oleh Raden Aprian Diaz Novandi dimana, kesimpulan yang dihasilkan adalah: "Algoritma Floyd-Warshall yang menerapkan pemrograman dinamis lebih menjamin keberhasilan

penemuan solusi optimum untuk kasus penentuan lintasan terpendek *all-pairs shortest path* [2].

Algoritma Flyod-Warshall adalah sebuah algoritma untuk mencari bobot minimum dan waktu tercepat dari graf berarah. Dalam satu kali eksekusi algoritma akan didapatkan jarak sebagai jumlah bobot dari lintasan terpendek antar setiap pasang simpul tanpa memperhitungkan informasi mengenai simpul-simpul yang dilaluinya, dengan kata lain Algoritma Floyd-Warshall adalah suatu metode yang melakukan pemecahan masalah dengan memandang solusi yang akan diperoleh sebagai suatu keputusan yang saling terkait, artinya solusi-solusi tersebut dibentuk dari solusi yang berasal dari tahap sebelumnya dan ada kemungkinan solusi lebih dari satu [2].

Kelebihan dari Algoritma Floyd-Warshall antara lain :

1. Algoritma Floyd-Warshall dapat digunakan untuk mencari jarak terpendek (shortest path) dari setiap pasangan node
2. Algoritma Floyd-Warshall menggunakan matriks bobot $n \times n$ sebagai masukan, dimana n merupakan jumlah verteks
3. Algoritma Floyd-Warshall dapat mentolerir *negative edge*.

Penelitian tentang penggunaan Algoritma Floyd-Warshall untuk mencari rute terpendek pernah dilakukan oleh sejumlah peneliti, antara lain: [3] menggunakan Algoritma Floyd-Warshall untuk mencari jalur terpendek pada penentuan tata letak parkir dan juga [4] tentang simulasi jaringan jalan di kota Semarang berbasis Algoritma Floyd- Warshall untuk menangani masalah lintasan terpendek.

Penelitian ini akan menggunakan *codebloks*untuk menghitung jarak (bobot) dari PT sampai ke tempat tujuan.

2. Landasan Teori

2.1. Distribusi

Distribusi adalah proses pemindahan atau dapat pula penyimpanan produk dari tingkat leveransir hingga sampai pada pengguna akhir dalam suatu rantai pasokan. Distribusi terjadi di antara setiap pasangan proses dalam rantai pasokan. Distribusi dilakukan baik saat pemindahan bahan mentah dari leveransir menuju pabrik pengolahan, hingga menjadi produk yang siap dikirimkan pada pelanggan [5].

2.2. Graf

Sebuah graf G didefinisikan sebagai suatu himpunan berhingga tak kosong dari objek - objek yang disebut verteks (minimal tunggal) bersama - sama dengan suatu himpunan yang anggotanya adalah pasangan yang tak terurut dari verteks yang berbeda pada G yang disebut edge (mungkin kosong), dan dinotasikan G(V (G), E(G)). Himpunan verteks dari G dinotasikan dengan V (G) dan himpunan edge dinotasikan dengan E(G) [6].

2.3. Pencarian Rute Terpendek

Lintasan terpendek adalah lintasan minimum yang diperlukan untuk mencapai suatu tempat dari tempat tertentu. Lintasan minimum yang dimaksud dapat dicari dengan menggunakan graf. Graf adalah sekumpulan titik di dalam bidang dua dimensi yang dihubungkan dengan sekumpulan edge. Sebuah graf dibentuk dari kumpulan titik yang dihubungkan dengan edges. Ada beberapa istilah yang berhubungan dengan graf, antara lain :

1. Titik - titik tersebut disebut verteks.
2. Garis - garis yang menghubungkan antar verteks disebut edge.
3. *Adjacent* artinya bertetangga. Maksudnya jika ada dua vertex disebut *adjacent*, jika mempunyai edge yang sama.
4. Path adalah lintasan yang melalui edge dan verteks dalam graf.
5. Cycle adalah lintasan yang dimulai dan berakhir pada verteks yang sama.
6. Direct pada directed graf adalah graf dimana edge-edgenya mempunyai suatu arah.

Graf yang digunakan adalah graf-graf yang berbobot, yaitu graf yang setiap sisinya diberikan suatu nilai. Ada beberapa macam persoalan lintasan terpendek, antara lain :

1. Lintasan terpendek antara dua buah simpul tertentu (*a pair shortest path*).
2. Lintasan terpendek antara semua pasangan simpul (*all pairs shortest path*).
3. Lintasan terpendek dari verteks tertentu ke semua verteks yang lain (*single source shortest path*).

4. Lintasan terpendek antara dua buah verteks yang melalui beberapa verteks tertentu (*intermediate shortest path*) [7].

2.4. Algoritma Floyd-Warshall

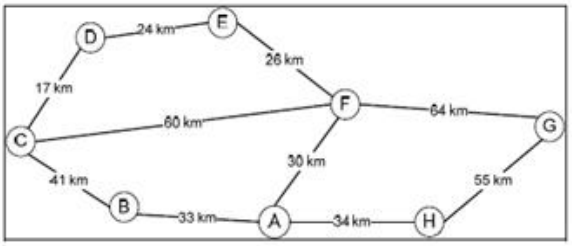
Algoritma Floyd-Warshall adalah salah satu varian dari pemrograman dinamis, yaitu suatu metode yang melakukan pemecahan masalah dengan memandang solusi yang akan diperoleh sebagai suatu keputusan yang saling terkait. Misalkan $d_{ij}^{(k)}$ menjadi bobot dari sebuah jalur terpendek dari verteks i ke verteks j dimana sebuah titik tengah di himpunan (1,2,...,k). Ketika k = 0, sebuh jalur dari verteks i ke verteks j dengan tidak adanya verteks hubung yang di jumlahkan lebih tinggi daripada 0 yang tidak memiliki semua verteks-verteks hubung. Seperti sbuah jalur yang memiliki satu sisi, dan kemudian $d_{ij}^{(0)} = w_{ij}$ [8].

$$d_{ij}^{k} = \begin{cases} w_{ij}, & \text{if } k = 0 \\ \min\left(d_{ij}^{(k-1)}, d_{ij}^{(k-1)}\right), & \text{if } k \geq 1 \end{cases}$$

Keterangan : d_{ij}^{k} menyatakan bobot Shortest path dari i ke j yang melalui titik ke (1,2,...., k) dan w_{ij} menyatakan sebuah adjacent. Algoritma ini bekerja denganmenghitung shortest path (i, j, k) untuk semua pasangan (i, j), kemudian hasil tersebut akan digunakan untuk menghitung shortest path (i, j, k) untuk semua pasangan (i, j), dst. Proses ini akan terus berlangsung hingga k = n dan kita telah menemukan jalur terpedek untuk semua pasangan (i,j) menggunakan simpul-simpul perantara [8].

Contoh kasus Algoritma Floyd-Warshall :

Untuk lebih jelasnya berikut adalah contoh dari Algoritma Floyd-Warshall: Misalkan terdapat suatu graf berbobot didefenisikan sebagai jarak yang merep-resentasikan kondisi keterhubungan antarkota di suatu daerah, dengan ilustrasi sebagai berikut:



A ke kota C. Orang tersebut mencoba untuk menerapkan Algoritma Floyd- Warshall dalam memilih rute terpendek didalam perjalanannya.

Untuk memperoleh jalur terpendek dengan menggunakan Algoritma Floyd- Warshall akan membuat beberapa tahap.

Dengan :

d_{ik} : nilai titik awal d_{kj}

: nilai titik tujuan

d_{ij} : nilai jarak sebenarnya.

Tahap-tahap pengerjaan :

1. Membuat tabel matriks jarak setiap titik.
2. Membuat tabel matriks jalur yang ingin dilewati .
3. Jika hasil penjumlah d_{ik} dan d_{kj} lebih kecil dari d_{ij}, maka ganti nilai d_{ij} menjadi $d_{ij} = d_{ik} + d_{kj}$.
4. Lakukan berulang sampai sebanyak verteksnya.
5. Menentukan jarak terpendek dari hasil perhitungan dengan :

$$d^k = \begin{cases} w_{ij}, & \text{if } k = 0 \\ \min(d_{ij}^{(k-1)}, d_{ij}^{(k-1)}), & \text{if } k \geq 1 \end{cases}$$

k = 0

	A	B	C	D	E	F	G	H
A	0	33	-	-	-	30	-	34
B	33	0	41	-	-	-	-	-
C	-	41	0	17	-	60	-	-
D	-	-	17	0	24	-	-	-
E	-	-	-	24	0	26	-	-
F	30	-	60	-	26	0	64	-
G	-	-	-	-	-	64	0	55
H	34	-	-	-	-	-	55	0

	A	B	C	D	E	F	G	H
A	A	B	C	D	E	F	G	H
B	A	B	C	D	E	F	G	H
C	A	B	C	D	E	F	G	H
D	A	B	C	D	E	F	G	H
E	A	B	C	D	E	F	G	H
F	A	B	C	D	E	F	G	H
G	A	B	C	D	E	F	G	H
H	A	B	C	D	E	F	G	H

k = 1

	A	B	C	D	E	F	G	H
A	0	33	-	-	-	30	-	34
B	33	0	41	-	-	63	-	67
C	-	41	0	17	-	60	-	-
D	-	-	17	0	24	-	-	-
E	-	-	-	24	0	26	-	-
F	30	63	60	-	26	0	64	64
G	-	-	-	-	-	64	0	55
H	34	67	-	-	-	64	55	0

	A	B	C	D	E	F	G	H
A	A	B	C	D	E	F	G	H
B	A	B	C	D	E	A	G	A
C	A	B	C	D	E	F	G	H
D	A	B	C	D	E	F	G	H
E	A	B	C	D	E	F	G	H
F	A	A	C	D	E	F	G	A
G	A	B	C	D	E	F	G	H
H	A	A	C	D	E	A	G	H

k = 2

	A	B	C	D	E	F	G	H
A	0	33	74	-	-	30	-	34
B	33	0	41	-	-	63	-	67
C	74	41	0	17	-	60	-	108
D	-	-	17	0	24	-	-	-
E	-	-	-	24	0	26	-	-
F	30	63	60	-	26	0	64	64
G	-	-	-	-	-	64	0	55
H	34	-	-	-	-	-	55	0

	A	B	C	D	E	F	G	H
A	A	B	B	D	E	F	G	H
B	A	B	C	D	E	A	G	A
C	B	B	C	D	E	F	G	B
D	A	B	C	D	E	F	G	H
E	A	B	C	D	E	F	G	H
F	A	A	C	D	E	F	G	A
G	A	B	C	D	E	F	G	H
H	A	A	C	D	E	A	G	H

Hasil

	A	B	C	D	E	F	G	H
A	0	33	74	-	-	30	-	34
B	33	0	41	-	-	63	-	67
C	74	41	0	17	-	60	-	108
D	-	-	17	0	24	-	-	-
E	-	-	-	24	0	26	-	-
F	30	63	60	-	26	0	64	64
G	-	-	-	-	-	64	0	55
H	34	-	-	-	-	-	55	0

	A	B	C	D	E	F	G	H
A	A	B	B	D	E	F	G	H
B	A	B	C	D	E	A	G	A
C	B	B	C	D	E	F	G	B
D	A	B	C	D	E	F	G	H
E	A	B	C	D	E	F	G	H
F	A	A	C	D	E	F	G	A
G	A	B	C	D	E	F	G	H
H	A	A	C	D	E	A	G	H

Maka dari hasil pencarian jalur terpendek dari A ke C menggunakan Algoritma Floyd-Warshall ditemukan bahwa jarak terpendek dari A ke C adalah 74 km dengan jalur (A- B - C).

Code::Bloks Application (C++)

Code::Blocks adalah suatu program lingkungan pengembangan terpadu bebas, nirlaba, bersumber terbuka dan lintas platform. Program yang ditulis dalam C++ beserta wxWidgets untuk GUI-nya ini bisa digunakan bersama dengan berbagai macam kompilator, contohnya GCC dan Visual C++.

3. Metodologi

Dalam penelitian ini data diperoleh dari PT. Rapy Ray Putratama yaitu data rute penjualan yang akan didistribusikan. Penelitian ini dilakukan selama dua bulan.

Jenis penelitian yang akan dilakukan dalam penelitian ini adalah studi kasus. Mengukur jarak yang ditempuh menggunakan *WikiMapia* seperti *Google Map* atau *Wez* .

Adapun langkah-langkah yang digunakan dalam penelitian ini adalah :

1. Mengumpulkan data jarak antara PT dengan outlet dan juga data antar outlet ke outlet.
2. Mengumpulkan Konsep-konsep dan teori-teori yang mendukung pemecahan masalah.
3. Mengumpulkan data pendistribusian barang selama kurang lebih 2 bulan.
4. Memilih outlate yang akan menjadi verteks pada graf.

5. Menentukan jalan yang akan digunakan menjadi edge pada graf.
6. Menyajikan persoalan tersebut ke dalam bentuk matriks yang berisi jarak antar outlet.
7. Menggambarkan graf jalan.
8. Mencari lintasan terpendek dengan menggunakan Algoritma Floyd-Warshall pada graf yang sudah di tentukan menggunakan Aplikasi Codeblocks.
9. Membuat Kesimpulan.

4. Hasil Dan Pembahasan

4.1. Menghitung Jarak Dengan Menggunakan Algoritma Floyd-Warshall

Dalam menghitung *Shortest Path*dengan menggunakan Algoritma Floyd- Warshall langkah yang harus di lakukan sebagai berikut :

1. Membuat tabel matriks.
2. Membuat tabel matriks rute yang ingin dilewati.
3. Jika hasil penjumlahan d_{ik} dan d_{kj} lebih kecil dari d_{ij}, maka ganti nilai d_{ij} menjadi $d_{ij} = d_{ik} + d_{kj}$
4. Lakukan berulang sampai sebanyak verteksnya (langkah ini menggunakan aplikasi Codeblocks)
5. Menentukan path terpendek dari hasil perhitungan.

Tabel 1. Jarak yang Belum menggunakan Algoritma Floyd Warshall

No	Jarak	Jalur yang Dilalui (sirkuit)	Panjang Jalan (km)
1	P-v_1	P-v_{31}-v_{20}-v_1-v_{20}-v_{31}-P	6.03
2	P-v_2	P-v_7-v_2-v_7-P	9.86
3	P-v_3	P-v_{33}-v_3-v_{33}-P	6.3
4	P-v_4	P-v_{38}-v_4-v_{38}-P	5.62
5	P-v_5	P-v_{17}-v_{10}-v_{41}-v_5-v_{41}-v_{10}-v_{17}-P	34.22
6	P-v_6	P-v_7-v_6-v_7-P	5.36
7	P-v_8	P-v_{33}-v_3-v_{21}-v_{12}-v_8-v_{12}-v_{21}-v_3-v_{33}-P	41.2
8	P-v_9	P-v_{33}-v_{37}-v_{11}-v_9-v_{11}-v_{37}-v_{33}-P	12.3
9	P-v_{10}	P-v_{17}-v_{10}-v_{17}-P	13.06
10	P-v_{11}	P-v_{33}-v_{37}-v_{11}-v_{37}-v_{33}-P	8.66
11	P-v_{12}	P-v_{33}-v_3-v_{21}-v_{12}-v_{21}-v_3-v_{33}-P	30.94
12	P-v_{13}	P-v_{15}-v_{36}-v_{39}-v_{13}-v_{39}-v_{36}-v_{15}-P	8.03
13	P-v_{14}	P-v_7-v_2-v_{14}-v_2-v_7-P	16.92
14	P-v_{16}	P-v_{25}-v_{40}-v_{24}-v_{16}-v_{24}-v_{40}-v_{25}-P	10.238
15	P-v_{18}	P-v_{15}-v_{30}-v_{18}-v_{30}-v_{15}-P	8.868
16	P-v_{19}	P-v_{33}-v_{37}-v_{34}-v_{19}-v_{34}-v_{37}-v_{33}-P	21.9
17	P-v_{20}	P-v_{31}-v_{20}-v_{31}-P	4.972
18	P-v_{21}	P-v_7-v_2-v_{21}-v_2-v_7-P	26.28
19	P-v_{22}	P-v_7-v_2-v_{22}-v_2-v_7-P	20.7
20	P-v_{23}	P-v_{38}-v_4-v_{16}-v_{14}-v_{23}-v_{14}-v_{16}-v_4-v_{38}-P	44.72
21	P-v_{24}	P-v_{38}-v_4-v_{40}-v_{24}-v_{40}-v_4v_{38}-P	10.508
22	P-v_{26}	P-v_{33}-v_{37}-v_{11}-v_9-v_{26}-v_9-v_{11}-v_{37}-v_{33}-P	21.28
23	P-v_{27}	P-v_{31}-v_{20}-v_1-v_{27}-v_1-v_{20}-v_{31}-P	10.71
24	P-v_{28}	P-v_{33}-v_{37}-v_{11}-v_9-v_{28}-v_9-v_{11}-v_{37}-v_{33}-P	17
25	P-v_{29}	P-v_7-v_{29}-v_7-P	4.338
26	P-v_{30}	P-v_{15}-v_{30}-v_{15}-P	3.828
27	P-v_{32}	P-v_{31}-v_{20}-v_{32}-v_{20}-v_{31}-P	5.816
28	P-v_{34}	P-v_{33}-v_{37}-v_{34}-v_{37}-v_{33}-P	11.58
29	P-v_{35}	P-v_7-v_{35}-v_7-P	3.758
30	P-v_{36}	P-v_{15}-v_{36}-v_{15}-P	3.314
31	P-v_{37}	P-v_{33}-v_{37}-v_{33}-P	6.38
32	P-v_{39}	P-v_{15}-v_{36}-v_{39}-v_{36}-v_{15}-P	6.254
33	P-v_{40}	P-v_{38}-v_{25}-v_{40}-v_{25}-v_{38}-P	8.138
34	P-v_{41}	P-v_{17}-v_{10}-v_{41}-v_{10}-v_{17}-P	18.28

Dengan menggunakan Algoritma Floyd-Warshall maka didapat hasil seperti tabel dibawah :

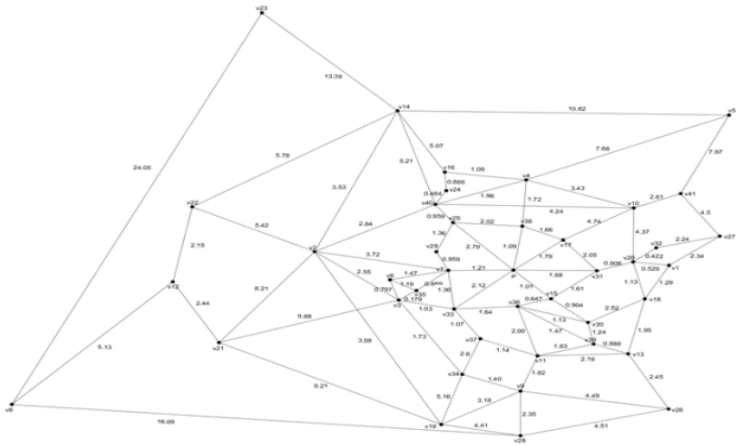
Tabel 2. Perhitungan Jarak dengan Menggunakan Algoritma Floyd-Warshall

No	Jarak	Jalur yang Dilalui (sirkuit)	Panjang Jalan (km)
1	P-v1	P-v31-v20-v1-v20-v31-P	6.03
2	P-v2	P-v7-v2-v7-P	9.86
3	P-v3	P-v33-v3-v33-P	6.3
4	P-v4	P-v38-v4-v38-P	5.62
5	P-v5	P-v38-v4-v5-v4-v38-P	20.98
6	P-v6	P-v7-v6-v7-P	5.36
7	P-v8	P-v7-v2-v22-v12-v8-v12-v22-v2-v7-P	35.26
8	P-v9	P-v15-v36-v11-v9-v11-v36-v15-P	10.954
9	P-v10	P-v38-v4-v10-v4-v38-P	12.48
10	P-v11	P-v15-v36-v39-v11-v39-v36-v15-P	9.914
11	P-v12	P-v7-v2-v22-v12-v22-v2-v7-P	24.5
12	P-v13	P-v15-v36-v39-v13-v39-v36-v15-P	8.03
13	P-v14	P-v7-v2-v14-v2-v7-P	16.92
14	P-v16	P-v38-v4-v16-v4-v38-P	7.8
15	P-v18	P-v31-v20-v18-v20-v31-P	7.232
16	P-v19	P-v7-v35-v3-v34-v9-v19-v9-v34-v3-v35-v7-P	17.02
17	P-v20	P-v31-v20-v31-P	4.972
18	P-v21	P-v33-v3-v21-v3-v33-P	26.06
19	P-v22	P-v7-v2-v22-v2-v7-P	20.7
20	P-v23	P-v7-v2-v14-v23-v14-v2-v7-P	43.7
21	P-v24	P-v25-v40-v24-v40-v25-P	8.466
22	P-v26	P-v15-v36-v39-v13-v26-v13-v39-v36-v15-P	12.93
23	P-v27	P-v31-v20-v1-v27-v1-v20-v31-P	10.71
24	P-v28	P-v15-v36-v11-v9-v28-v9-v11-v36-v15-P	15.654
25	P-v29	P-v7-v29-v7-P	4.338
26	P-v30	P-v15-v30-v15-P	3.828
27	P-v32	P-v31-v20-v32-v20-v31-P	5.816
28	P-v34	P-v33-v3-v34-v3-v33-P	9.76
29	P-v35	P-v7-v35-v7-P	3.758
30	P-v36	P-v15-v36-v15-P	3.314
31	P-v37	P-v33-v37-v33-P	6.38
32	P-v39	P-v15-v36-v39-v36-v15-P	6.254
33	P-v40	P-v25-v40-v25-P	7.498
34	P-v41	P-v38-v4-v10-v41-v10-v4-v38-P	17.7

Dari tabel diatas didapatkan jarak minimum setiap outlet dengan menggunakan rumus :

$$d^k = \begin{cases} w_{ij}, & \text{if } k = 0 \\ \min(d_{ij}^{(k-1)}, d_{ij}^{(k-1)}), & \text{if } k \geq 1 \end{cases}$$

Adapun gambar graf yang telah dicari :



Cara mencari rute terpendek dengan menggunakan Codeblocks. Adapun *pseucode* :

```
#include<iostream>
#include<conio.h>
using namespace std;
void floyds(double d[][42])
{
    int i,j,k;
    for(k=0;k<42;k++)
    {
        for(i=0;i<42;i++)
        {
            for(j=0;j<42;j++)
            {
                if((d[i][k]*d[k][j]!=0)&&(i!=j))
                {
                    if((d[i][k]+d[k][j]<d[i][j])||(d[i][j]==0))
                    {
                        d[i][j]=d[i][k]+d[k][j];
                    }
                }
            }
        }
```

```
    }
    for(i=0;i<42;i++)
    {
        cout<<"\nminimum Cost With Respect to node:"<<i<<endl;
        for(j=0;j<42;j++)
        {
            cout<<d[i][j]<<"\t";
        }
    }
}
int main()
{
    double d[42][42];
    cout<<"ENTER VELUES OF ADJACENCY MATRIX\n\n";
    for(int i=0;i<42;i++)
    {
        cout<<"enter values for "<<(i+1)<<" row"<<endl;
        for(int j=0;j<42;j++)
        {
            cin>>d[i][j];
        }
    }
    floyds(d);
    getch();
}
```

Dalam menggunakan *Codeblocks* data yang dimasukkan adalah data dari matriks baris atau data jaraknya. Output yang akan keluar sebagai berikut :

```
minimum Cost With Respect to node:0
0       3.015   4.608   2.058   2.81    10.49   2.68    1.21
17.308  5.188   6.24    3.657   12.178  4.015   8.138   1.01
3.9     1.79    3.616   8.188   2.486   11.938  10.828  21.528
4.233   2.79    6.465   5.148   7.538   2.169   1.914       1.68
2.908   2.12    3.788   1.879   1.657   3.19
1.89    3.127   3.749   8.85
minimum Cost With Respect to node:1
3.015   0       7.623   5.073   5.825   13.505  5.695   4.225
20.323  7.25    4.899   5.43    15.193  3.24    11.153      2.045
6.915   3.385   1.29    10.43   0.529   14.953      13.843  24.543
7.248   5.805   5.69    2.34    9.6
5.184   3.81    1.335   0.951   5.135   6.803   4.894   3.592
6.205   4.105   4.128   6.764   6.84
minimum Cost With Respect to node:2
4.608   7.623   0       2.55    4.8     12.48   3.347   3.398
12.7    5.68    7.08    5.79    7.57    7.578   3.53
5.618   4.21    6.398   8.224   3.58    7.094   8.21    5.42
16.92   3.324   3.799   10.028  9.756   7.99    4.357   6.35
6.288   7.516   3.58    4.28    2.729   5.22    4.65    5.698
6.69    2.84    9.69
minimum Cost With Respect to node:3
2.858   5.073   2.55    0       4.868   12.548  0.797   0.848
15.25   3.13    8.298   3.24    10.12   5.028   6.08
3.068   5.496   3.848   5.674   6.13    4.544   9.88    7.97
19.47   4.61    3.167   7.478   7.206   5.48    1.807   3.8
3.738   4.966   1.03    1.73    0.179   2.67    2.1     3.148
4.14    4.126   10.908
minimum Cost With Respect to node:4
2.81    5.825   4.8     4.868   0       7.68    5.49    4.02
17.5    7.998   3.43    6.467   12.37   6.825   6.16    3.82
1.89    3.38    6.426   8.38    5.296   13.01   10.22   19.55
1.976   2.919   9.275   7.958   10.348  4.279   4.724       4.49
5.718   4.93    6.598   4.689   4.467   6
1.72    5.937   1.96    6.04
minimum Cost With Respect to node:5
10.49   13.505  12.48   12.548  7.68    0       13.17   11.7
23.88   15.678  10.58   14.147  18.75   14.505  10.82   11.5
8.77    11.06   14.106  16.06   12.976  20.69   16.6    24.21
9.656   10.599  16.955  12.47   18.028  11.959  12.404      12.17
13.398  12.61   14.278  12.369  12.147  13.68       9.4     13.617
9.64    7.97
```

Dari tabel diatas menjelasakan tentang lintasan yang memiliki bobot terendah di setiap verteksnya.

4.2. Persentase Penghematan

Jarak Tabel 3 Rekapitulasi data

	Rute	Perhitungan Penghematan			
		Jarak (km)		%	
		Reguler-Usulan	Selisih	$\frac{Jarak_{lama} - Jarak_{baru}}{Jarak_{lama}} \times 100\%$	Hasil
Jarak	P-v_5	34.22 - 20.98	13.24	$\frac{13.24}{34.22} \times 100\%$	38.69%
	P-v_8	41.2 - 35.26	5.94	$\frac{5.94}{41.2} \times 100\%$	14.42%
Pada	P-v_9	12.3 - 10.954	1.346	$\frac{1.346}{12.3} \times 100\%$	10.94%
	P-v_{10}	13.06 - 12.48	0.58	$\frac{0.58}{13.06} \times 100\%$	4.44%
	P-v_{12}	30.94 - 24.5	6.44	$\frac{6.44}{30.94} \times 100\%$	20.81%
	P-v_{16}	10.238 - 7.8	2.438	$\frac{2.438}{10.238} \times 100\%$	23.81%
Rute	P-v_{18}	8.868 - 7.232	1.636	$\frac{1.636}{8.868} \times 100\%$	18.45%
	P-v_{19}	21.9 - 17.02	4.88	$\frac{4.88}{21.9} \times 100\%$	22.28%
Outlet	P-v_{21}	26.28 - 26.06	0.22	$\frac{0.22}{26.28} \times 100\%$	0.84%
	P-v_{23}	44.72 - 43.7	1.02	$\frac{1.02}{44.72} \times 100\%$	2.28%
	P-v_{24}	10.508 - 8.466	2.042	$\frac{2.042}{10.508} \times 100\%$	19.43%
	P-v_{26}	21.28 - 12.93	8.35	$\frac{8.35}{21.28} \times 100\%$	39.24%
	P-v_{28}	17 - 15.654	1.346	$\frac{1.346}{17} \times 100\%$	7.92%
	P-v_{34}	11.58 - 9.76	1.82	$\frac{1.82}{11.58} \times 100\%$	15.72%
	P-v_{40}	8.138 - 7.498	0.64	$\frac{0.64}{8.138} \times 100\%$	7.86%
	P-v_{41}	18.28 - 17.7	0.58	$\frac{0.58}{18.28} \times 100\%$	3.17%

Dari tabel rekapitulasi di atas dapat diketahui bahwa :

1. Perbandingan jarak antara rute Reguler dengan rute Floyd-Warshall menunjukkan bahwa hasil perhitungan dengan metode Algoritma Floyd-Warshall memiliki hasil jarak yang lebih pendek.

2. Perbandingan jarak yang memiliki tingkat perbedaan persentase dalam penghematan.

Dengan menggunakan rumus yang sama maka total persentase penghe-matan jarak yang didapat adalah 10.97% (51.304 km).

Data yang berada dioutput pada lampiran B adalah angka jarak pada masing-masing titik yang menjelaskan jarak yang paling minimum. Prosesnya dilakukan dengan memasukkan fungsi seperti dilampiran A. Contoh pada iterasi 0 di lampiran B, ada 42 jumlah jarak minimum, dimulai dari 0 yang berarti titik dimulai dari P ke P, selanjutnya 3.015 km, 4.608 km, 2.058 km, dan seterusnya. jarak 3.015 km yang berarti dari titik P ke titik v_1 adalah yang paling minimum.

5. Kesimpulan aan Saran

5.1. Kesimpulan

Berdasarkan pengolahan data maka diperoleh kesimpulan, yaitu dalam menentukan rute terpendek dengan mengunakan Algoritma Floyd-Warshall dengan 42 verteks yang telah dipilih maka PT. Rapy Ray Putratama sudah dapat menentukan jalur yang lebih pendek yang telah peneliti cari pada penelitian kali ini.

Pembentukan rute usulan yang dihasilkan dengan menggunakan metode Algoritma Floyd-Warshall menghasilkan rute yang lebih dekat dengan total jarak penghematan adalah 10.97 % (51.304 km).

5.2. Saran

Adapun saran-sarannya adalah sebagai berikut :

1. Untuk memperdalam dan mengembangkan wawasan disiplin ilmu yang telah dipelajari untuk mengkaji permasalahan tentang rute terpendek.
2. Bagi penelitian selanjutnya agar mempertimbangkan waktu dan juga kapasitas angkutan.
3. Mengembangkan aplikasi Algoritma Floyd-Warshall untuk menyelesaikan persoalan pencarian jalur terpendek yang lain.
4. Kepada PT. Rapy Ray Putratama Medan untuk menggunakan Algoritma Floyd-Warshall beserta rute yang telah didapatkan dalam menentukan rute terpendek yang akan dilewati driver PT. Rapy Ray Putratama Medan dalam mendistribusikan barang.

DAFTAR PUSTAKA

[1] Kakay, T. J., (2008): *Pemrograman Linier*, ANDI, Yogyakarta.

[2] Novandi, R. A. D., (2007): *Perbandingan Algoritma Dijkstra dan Algoritma Floyd-Warshall dalam Penentuan Lintasan Terpendek (Single Shortest Path)*, Institut Teknologi Bandung, Bandung.

[3] N. K. D. A. Jayanti, (2014). Penggunaan Algoritma Floyd-Warshall Dalam Masalah Jalur Terpendek Pada Penentuan Tata Letak Parkir. *Seminar Nasional Informatika*. Denpasar. 43,

[4] Harsono, dkk. (2016). Simulasi Jaringan Jalan di Kota Semarang Berbasis Algoritma Floyd-Warshall Untuk Menangani Masalah Lintasan Terpendek. *Unnes Journal of Mathematics.* Vol. 5, No. 2, 153-160.

[5] Chopra, S., dan Meind, P., (2016): *Supply Chain Management*. *Strategy,Planning and Operation*, Sixth Edition, Pearson Education,Inc, United States of America.

[6] Chartrand G, Lesnia L, d. Z. P., (2016): *Graphs And Digraphs*, 6rd Edition, Chapman And Hall/CRC, London

[7] Wilson, R. J., (2010): *Pengantar Teori Graf*, Edisi Kelima, Erlangga, Jakarta.

[8] Cormen, T. H., Leiserson, C. E., Rivest, R. L., dan Stein, C., (2001): *Intoduction To Algorithms*, Second Edition, McGraw-Hill, America.