

# PENENTUAN JALUR LINTASAN TERPENDEK TAKSI STAR CAB MENUJU TEMPAT WISATA DI KOTA PALEMBANG DENGAN ALGORITMA DIJKSTRA

Deri Susanti

Jurusan Sistem Informasi, STMIK Palcomtech, Palembang

E-mail : deri\_susanti@palcomtech.ac.id

**Abstrak** - Penggunaan algoritma djikstra untuk menentukan jalur terpendek pada lintasan antar *node* untuk mencapai tujuan akhir dengan jarak yang singkat, sehingga algoritma ini dapat menyelesaikan masalah untuk penentuan jalur terpendek. Hal ini sangat bermanfaat bagi wisatawan dan sopir taksi *Star Cab* Kota Palembang, sehingga dapat menghindari titik kemacetan dan dapat melalui jarak terpendek sehingga dapat mempersingkat perjalanan menuju tempat wisata yang dituju di Kota Palembang dengan biaya argo yang minimal. Dengan menggunakan algoritma *dijkstra* untuk menentukan lintasan terpendek menuju tempat wisata di Kota Palembang dengan menggunakan taksi *Star Cab* sebagai objek kendaraannya yang di uji dan diawali dengan menentukan jarak antar *node* dari *node* awal keberangkatan menuju *node* tujuan, maka setiap *node* dapat dihitung jaraknya, seperti menentukan jarak antar *node* awal dengan *node* terdekat setelahnya, kemudian setelahnya hitung kembali jarak antar *node* berikutnya setiap *node* yang telah dilewati tidak dihitung lagi sampai setiap *node* sudah terpenuhi. *Node-node* yang digunakan dalam implementasi pengujian Algoritma Dijkstra adalah *node* awal keberangkatan taksi *Star cab* menuju penjemputan wisatawan dengan tujuan akhir tempat wisata yang ada di pusat Kota Palembang. Pada penelitian ini pengujian algoritma *Dijkstra* menggunakan Bahasa pemrograman C++ sehingga dari hasil pengujian ini menghasilkan jalur lintasan terpendek sejauh 5 km dengan jalur A, C, E, F. hal ini dapat mempersingkat waktu perjalanan dan menghemat biaya ongkos argo meter pada taksi *Star Cab*

**Kata kunci** - Jalur Terpendek, Algoritma, Dijkstra, Taksi Star Cab, Tempat Wisata

## I. PENDAHULUAN

Taksi *Star Cab* merupakan taksi minibus yang dikembangkan oleh PT. Asoka Puri Kreasi yang bergerak dibidang jasa angkutan umum. Untuk saat ini kebutuhan akan transportasi umum di Kota Palembang meningkat dengan pesat oleh Karena itu PT. Asoka Puri Kreasi menghadirkan Taksi *Star Cab* menggunakan argo meter dengan harga yang terjangkau sehingga dapat memberikan kepuasan, keamanan dan kenyamanan kepada masyarakat Kota Palembang.

Kota Palembang adalah Ibu Kota Propinsi Sumatera Selatan. Kota Palembang saat ini sedang dalam proses pembangunan Light Rail Transit (LRT), hal ini menimbulkan kemacetan kendaraan di mana-mana terutama arah Jembatan Ampera. Jembatan Ampera merupakan icon atau identitas yang identik dengan Kota Palembang yang

selalu ramai dikunjungi oleh wisatawan baik wisatawan lokal maupun wisatawan luar negeri. Selain Jembatan Ampera adalah tempat wisata yang dapat dikunjungi oleh wisatawan yang mana titik awal keberangkatannya dimulai dari jembatan ampera yang tujuan berikutnya adalah tempat wisata yang terletak di Kota Palembang seperti ke Pulau Kemaro. Untuk menuju ke Pulau Kemaro harus menyeberangi sungai musi dengan menaiki perahu kecil yang disebut dengan “ketek” oleh masyarakat setempat. Untuk penyewaan ketek itu sendiri bisa kita dapatkan dibawah jembatan ampera yang berdekatan dengan pasar 16 Palembang.

Permasalahan transportasi pada kota besar yang mempunyai jaringan transportasi yang rumit umumnya sama. orang sering mengetahui rute perjalanan ke tempat yang biasa di kunjunginya, akan tetapi jika tempat tujuan tersebut belum pernah dikunjungi rata - rata mereka sering kesulitan untuk menentukan rute untuk mencapai tempat tersebut (Chiu et al, 2005) dalam Arifianto (2012)<sup>[1]</sup>. selama ini orang akan bertanya kepada orang lain yang mengetahui betul jaringan transportasi di kota tersebut

Algoritma *dijkstra* dapat membantu dalam penyelesaian jalur terpendek untuk menuju tempat wisata yang mana titik tujuannya yaitu Jembatan Ampera dari jembatan Ampera kemudian penyebrangan dilakukan dengan transportasi air menuju pulau kemaro oleh Karena itu maka diambil jembatan Ampera sebagai titik tuuan akhirnya. Hal ini sangat bermanfaat untuk wisatawan dan sopir taksi *Star Cab* dalam menentukan rute terpendek dalam sistem operasinya dan menghindari titik-titik macet Kota Palembang. Hal ini dapat menghemat biaya BBM untuk transportasi dan lebih cepat sampai ketujuan, dan hal ini juga sangat membantu dan meringankan biaya sewa konsumen.

Menurut Pradana dalam Kriwanto (2014) <sup>[3]</sup> yaitu lintasan terpendek suatu graf. Algoritma *dijkstra* hanya dapat digunakan pada graf yang berbobot positif sedangkan algoritma *bellman ford* dapat digunakan *graf* dengan bobot positif maupun negatif. Jika pada penelitian ini algoritma yang digunakan adalah algoritma *dijkstra* dan algoritma *bellman-ford*, maka dalam penelitian yang akan dilakukan yaitu membangun sistem menggunakan *algoritma floyd-warshall*. Selain itu jika pada penelitian ini mengimplementasikan sistem dengan permasalahan matematika, maka penelitian yang akan dilakukan mengimplementasikan pada persoalan transportasi umum.

Menurut Arifianto (2012) <sup>[1]</sup>. Tujuan penggunaan algoritma *dijkstra* dalam penelitian ini adalah untuk menentukan jalur terpendek berdasarkan bobot terkecil dari satu *node* ke *node* yang lain dan jumlah *node* adalah penentuan dari keefektifan dari algoritma *dijkstra* tersebut.

Untuk menentukan rute terpendek dengan penggunaan algoritma *dijkstra* dapat dilakukan dengan beberapa tahap, seperti :

- Untuk setiap jarak antar *node* diberi nilai bobotnya.
- Tentukan *node* awal sebagai keberangkatan dan tentukan *node-node* yang akan dilalui kendaraan
- Dari *node* awal hitung setiap *node* yang akan dilalui untuk menuju *node* tujuan
- Setelah setiap *node* diberi nilai dengan jarak *node* yang akan ditempuh, dan *node* yang sudah diberi nilai tidak dicek kembali dan jarak yang terpendek adalah jarak terakhir yang mempunyai nilai bobotnya terkecil atau nilai minimal.
- Tentukan *node* yang belum dihitung nilai bobotnya dengan jarak terkecil (*node* awal) sebagai “*node* keberangkatan” berikutnya. Untuk menentukan *node* selanjutnya dapat dimlai lagi dari langkah c.

Menurut Aditya Pradhana Bayu (2013) dalam Priatmoko<sup>[4]</sup> Algoritma *Djiksra* ditemukan oleh Edsger.Wybe *Djiksra* pada tahun 1959. Algoritma ini merupakan algoritma yang dapat memecahkan masalah pencarian jalur terpendek dari suatu graf pada setiap simpul yang bernilai tidak negatif. *djiksra* merupakan algoritma yang termasuk dalam algoritma *greedy*, yaitu algoritma yang sering digunakan untuk memecahkan masalah yang berhubungan dengan suatu optimasi. Dalam pencarian jalur terpendeknya algoritma *djiksra* bekerja dengan mencari bobot yang paling minimal dari suatu graf berbobot, jarak terpendek akan diperoleh dari dua atau lebih titik dari suatu graf dan nilai total yang didapat adalah yang bernilai paling kecil.

Misalkan  $G$  adalah graf berarah berlabel dengan titik-titik  $V(G) = \{v_1, v_2, \dots, v_n\}$  dan path terpendek yang dicari adalah dari  $v_1$  ke  $v_n$ . Algoritma *Djiksra* dimulai dari titik  $v_1$ . Dalam iterasinya, algoritma akan mencari satu titik yang jumlah bobotnya dari titik 1 terkecil. Titik-titik yang terpilih dipisahkan, dan titik-titik tersebut tidak diperhatikan lagi dalam iterasi berikutnya. Langkah-langkah dalam menentukan lintasan terpendek pada algoritma *Djiksra* yaitu:

- Pada awalnya pilih *node* sumber sebagai *node* awal, diinisialisasikan dengan „1“.
- Bentuk tabel yang terdiri dari *node*, status, bobot, dan *predecessor*. Lengkapi kolom bobot yang diperoleh dari jarak *node* sumber ke semua *node* yang langsung terhubung dengan *node* sumber tersebut.
- Jika *node* sumber ditemukan maka tetapkan sebagai *node* terpilih.
- Tetapkan *node* terpilih dengan label permanen dan perbaharui *node* yang langsung terhubung.
- Tentukan *node* sementara yang terhubung pada *node* yang sudah terpilih sebelumnya dan merupakan bobot terkecil dilihat dari tabel dan tentukan sebagai *node* terpilih berikutnya.
- Apakah *node* yang terpilih merupakan *node* tujuan?. Jika ya, maka kumpulan *node* terpilih atau *predecessor* merupakan rangkaian yang menunjukkan lintasan terpendek.

Menurut Satyananda (2012 : 46) dalam Solichin<sup>[5]</sup> menjelaskan bahwa Algoritma *Dijkstra* merupakan salah satu algoritma yang efektif dalam memberikan lintasan terpendek dari suatu lokasi ke lokasi yang lain. Prinsip dari algoritma *Dijkstra* adalah dengan pencarian dua lintasan yang paling kecil. Algoritma *Dijkstra* memiliki iterasi untuk mencari titik yang jaraknya dari titik awal adalah paling pendek. Pada setiap iterasi, jarak titik yang diketahui (dari titik awal) diperbarui bila ternyata didapat titik yang baru yang memberikan jarak terpendek. Syarat algoritma ini adalah bobot sisinya yang harus non-negatif.). Menurut Alfred V. Aho (1974) menjelaskan rincian algoritma *Dijkstra* sebagai prosedur sebagai berikut:

Input: graf terhubung dan berarah  $G=(V,E)$ , matriks bobot  $C$ , titik awal  $v_0$

Output: jarak terpendek dari titik  $V_0$  ke titik lain dalam  $D$

Prosedur *Dijkstra* ( $G, C, v_0$ )

Mulai

$S \leftarrow \{ v_0 \} ;$

$D[v_0] \leftarrow 0 ;$

Untuk masing-masing  $v$  dalam  $V - \{ v_0 \}$  lakukan

$D[v] \leftarrow C[v_0, v] ;$

Selama  $S \neq V$  lakukan

Mulai

Pilih salah satu titik  $w$  di  $v - s$  dimana  $D[w]$  adalah minimum;

$S \leftarrow S \cup \{ w \} ;$

Untuk masing-masing  $v$  di  $V - S$  lakukan

$D[v] \leftarrow \min (D[w], D[w] + C[w, v] ) ;$

Selesai

Selesai

Dalam penelitian ini dimisalkan *node* awalnya adalah *pool* taksi *Star Cab* sedangkan untuk *node* akhirnya adalah Jembatan Ampera sebagai *node* akhir tujuannya. Yang mana perjalanan yang ditempuh oleh Taksi *Star Cab* untuk menjemput wisatawan adalah dimulai dari Pool Taksi *Star Cab* kemudian Bandara Internasional Sultan Mahmud Badaruddin II dan Jembatan Ampera.

Sedangkan pembangunan *Light Rail Transit* (LRT) yang dimulai dari Bandara Sultan Mahmud Badaruddin II melewati jalan Gubernur H.Asnawi Mangku Alam, Jl.Tj.Api-Api, Jl.Lintas Sumatera, Jl. Kapten A.Rivai, Jl.Kol.Atmo, Jembatan Ampera, Jl.Jendral Ahmad Yani, Jl.Gub.H.Bari, Stadion Jaka Baring dan berakhir di Stasiun Bangunan Depo. Dan rute tersebut merupakan rute yang memiliki titik kemacetan yang paling tinggi saat ini. Oleh karen itu dengan menggunakan algoritma *Dijkstra* diharapkan mampu menghindari titik kemacetan tersebut. Berikut gambar dari proyek pembangunan LRT Palembang .



Sumber : Google

Gambar 1. Denah Lokasi Pembangunan LRT

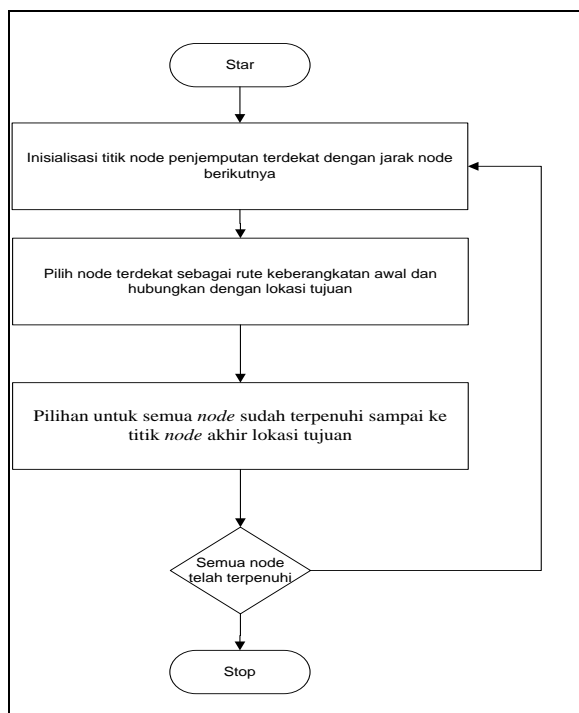
## II. METODOLOGI PENELITIAN

### 1. Bahan Penelitian

Objek yang dijadikan bahan penelitian ini adalah transportasi umum milik PT. Asoka Puri Kreasi yaitu taksi *starcab* yang merupakan transportasi umum yang menggunakan argo meter dengan tarif minimum

### 2. Alat Penelitian

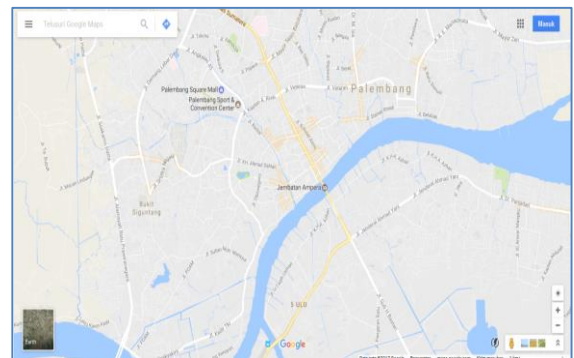
Metode penelitian ini dilakukan berdasarkan studi literatur yang merupakan pengembangan dari penelitian terdahulu dan pengujiannya menggunakan Bahasa pemrograman C++. Alur penelitian pada algoritma *dijkstra* dapat dilihat dari *flowchart* berikut ini :



Gambar 2. Flowchart Metodologi Penelitian

## III. HASIL DAN PEMBAHASAN

Berikut adalah tampilan Peta Kota Palembang Provinsi Sumatera Selatan. Yang mana dari gambar ini dapat kita analisis sebuah kajian untuk menentukan rute terpendek untuk menuju tempat wisata yang ada di Kota Palembang khususnya untuk menginisialisasi jarak antara *node* pertama, penjemputan wisatawan atau kosumen sampai ke tempat tujuan akhir tempat wisata, seperti Jembatan Ampera.



Sumber : Google Maps

Gambar 3. Peta Kota Palembang

Dari Peta Kota Palembang tersebut dapat dilihat jalan utama yang dapat dilewati oleh transportasi umum sehingga kita bisa menginisialisasi jarak dari rute-rute yang akan ditempuh untuk menghindari titik kemacetan yang saat ini terjadi di kota Palembang, berikut adalah data jarak perjalanan yang dimulai dari *Pool Taksi Star Cab* menuju titik penjemputan sampai ke titik tujuan akhir. Berikut adalah data jarak tempuh yang dapat dilalui oleh transportasi umum dengan menggunakan taksi *Star Cab*.

Tabel 1. Data Rute Kendaraan

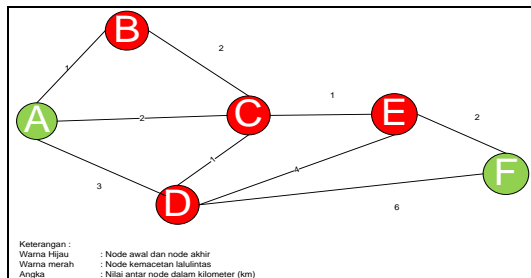
Rute		Jarak (Km)	Waktu (Menit)
Pool – Bandara – Jembatan Ampera	1 Jl.Kol.H.Burlian – Jl.Lintas Sumatera – Jln.Tj.Api-Api	12.6	26
	2 Jl.H.M. Nurdin Pandji	12.7	28
	3 Jl.H.M. Nurdin Pandji – Jl.Gubernur H.Asnawi Mangku alam	12.8	30
Pool-Bandara – Jembatan Ampera	4 Jl.Lintas Sumatera	15.6	38
	5 Jl.H.M. Nurdin Pandji - Jl.Lintas Sumatera	16.5	40
	6 Jl.H.M. Nurdin Pandji	15.1	42
Pool – Jembatan Ampera	7 Jl. Jend. Sudirman – Jl. Lintas Sumatera	5.1	16

	8	Jl. Demang Lebar Daun – Jl. Jend. Sudirman – Jl. Lintas Sumatera	5.8	17
	9	Jl. POM IX - Jl. Jend. Sudirman – Jl. Lintas Sumatera	6.1	19

#### A. Pembahasan

##### 1. Inisialisasi titik *node* penjemputan terdekat dengan jarak *node* berikutnya

pada pembahasannya jarak anatar *node* diinisialkan dalam perhitungannya dinyatakan dalam kilometer (km), sebagai sampel untuk perkiraan jarak anatar *nodenya* dapat dilihat pada gambar berikut :



Gambar 4. *Node-node* rute keberangkatan taksi *Star Cab*

Dari gambar 4 diatas bisa dijelaskan yaitu, jika pada *node* A adalah *node* awal untuk keberangkatan untuk menuju *node* terdekat berikutnya, maka untuk jarak atau rute keberangkatan dapat dihitung dengan rumus sebagai berikut :

Rumus :

$$V = \{ 1, 2, 3, 4 \}$$

$$E = \{ (1, 2), (2, 3), (1, 3), (1, 3), (2, 4), (3, 4), (3, 4), (3, 3) \}$$

$$= \{ e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8 \}$$

Inisialisasi seluruh *node* yang akan ditempuh dapat dilihat pada tabel. 2 dibawah in :

Tabel.2 Inisialisasi data jarak antar *node*

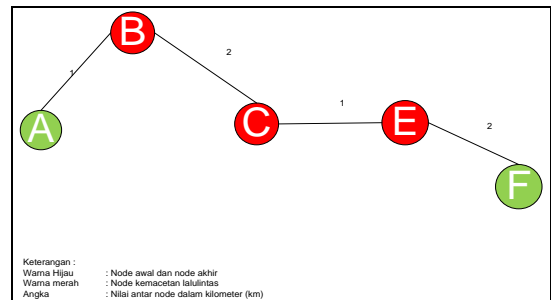
No	Node	Jarak antar node (km)	Jumlah Jarak antar node (km)
1	A → B → C → E → F	1 → 2 → 1 → 2	6
2	A → C → E → F	2 → 1 → 2	5
3	A → D → C → E → F	3 → 1 → 1 → 2	7
4	A → D → E → F	3 → 4 → 2	8
5	A → D → F	3 → 6	9

##### 2. Pilih *node* terdekat sebagai rute keberangkatan awal dan hubungan dengan lokasi tujuan

Maka untuk menentukan rute atau jarak terpendek dapat ditentukan dengan cara sebagai berikut :

###### a. Rute pertama

Berdasarkan gambar 4 maka akan diambil sampel pengujian untuk rute pertama adalah *node* A, B, C, E, dan F.



Gambar 5. *Node* rute keberangkatan pertama

$$V = \{ A, B, C, E, F \}$$

$$E = \{ (A, B) (B, C) (C, E) (E, F) \}$$

$$= \{ (1) (2) (1) (2) \}$$

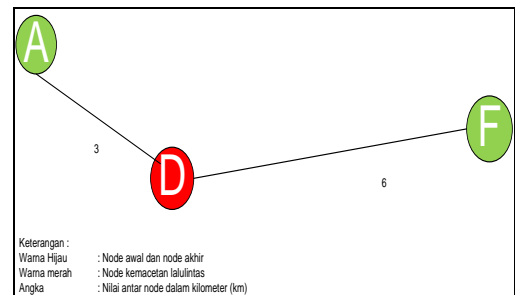
$$= \{ 1 + 2 + 1 + 2 \}$$

$$= 6 \text{ km}$$

Maka jarak antara *node* A → B → C → E → F dapat ditempuh dengan jarak sejauh 6 km.

###### b. Rute kedua

Berdasarkan gambar 4 maka akan diambil sampel pengujian untuk rute pertama adalah *node* A, D dan F.



Gambar 6. *Node* rute keberangkatan kedua

$$V = \{ A, D, F \}$$

$$E = \{ (A, D) (D, F) \}$$

$$= \{ (3) (6) \}$$

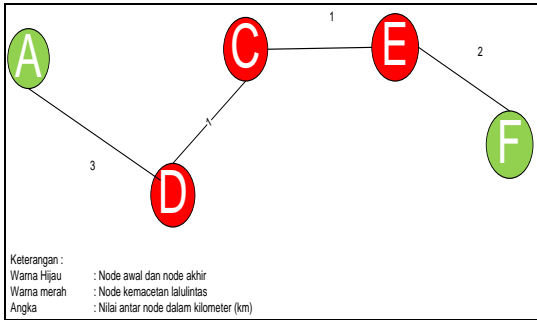
$$= \{ 3 + 6 \}$$

$$= 9 \text{ km}$$

Maka jarak antara *node* A → D → F dapat ditempuh dengan jarak sejauh 9 km.

###### c. Rute ketiga

Berdasarkan gambar 4 maka akan diambil sampel pengujian untuk rute pertama adalah *node* A, D, C, E, dan F.



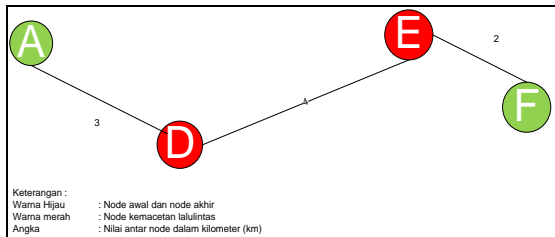
Gambar 7. Node rute keberangkatan ketiga

$$\begin{aligned} V &= \{A, D, C, E, F\} \\ E &= \{ (A, D) (D, C) (C, E) (E, F) \} \\ &= \{ (3) (1) (1) (2) \} \\ &= \{ 3 + 1 + 1 + 2 \} \\ &= 7 \text{ km} \end{aligned}$$

Maka jarak antara node  $A \rightarrow D \rightarrow C \rightarrow E \rightarrow F$  dapat ditempuh dengan jarak sejauh 7 km.

d. Rute keempat

Berdasarkan gambar 4 maka akan diambil sampel pengujian untuk rute pertama adalah *node* A, D, E, dan F.



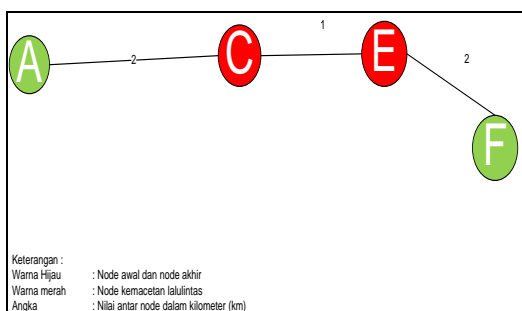
Gambar 8. Node rute keberangkatan keempat

$$\begin{aligned} V &= \{A, D, E, F\} \\ E &= \{ (A, D) (D, E) (E, F) \} \\ &= \{ (3) (4) (2) \} \\ &= \{ 3 + 4 + 2 \} \\ &= 9 \text{ km} \end{aligned}$$

Maka jarak antara node  $A \rightarrow D \rightarrow E \rightarrow F$  dapat ditempuh dengan jarak sejauh 9 km.

e. Rute kelima

Berdasarkan gambar 4 maka akan diambil sampel pengujian untuk rute pertama adalah *node* A, C, E, dan F.



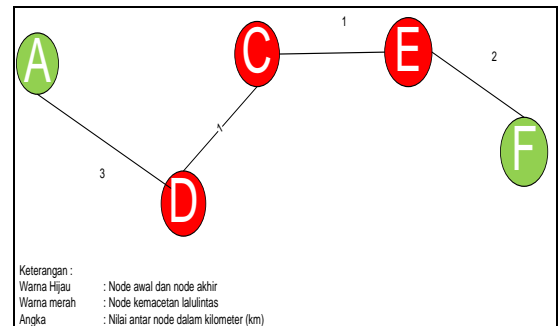
Gambar 9. Node rute keberangkatan kelima

$$\begin{aligned} V &= \{A, C, E, F\} \\ E &= \{ (A, C) (C, E) (E, F) \} \\ &= \{ (2) (1) (2) \} \\ &= \{ 2 + 1 + 2 \} \\ &= 5 \text{ km} \end{aligned}$$

Maka jarak antara node  $A \rightarrow C \rightarrow E \rightarrow F$  dapat ditempuh dengan jarak sejauh 5 km.

f. Rute keenam

Berdasarkan gambar 4 maka akan diambil sampel pengujian untuk rute pertama adalah *node* A, D, C, E, dan F.



Gambar 10. Node rute keberangkatan keenam

$$\begin{aligned} V &= \{A, D, C, E, F\} \\ E &= \{ (A, D) (D, C) (C, E) (E, F) \} \\ &= \{ (3) (1) (1) (2) \} \\ &= \{ 3 + 6 \} \\ &= 9 \text{ km} \end{aligned}$$

Maka jarak antara node  $A \rightarrow D \rightarrow C \rightarrow E \rightarrow F$  dapat ditempuh dengan jarak sejauh 9 km.

3. Pilihan untuk semua *node* sudah terpenuhi sampai ke titik *node* akhir lokasi tujuan

Dari pembahasan ke enam rute yang diujikan maka dapat disimpulkan bahwa *node* untuk lintasan terpendek terdapat pada perhitungan rute ke dua yaitu 5 km lebih cepat dari rute yang lainnya.

B. Hasil

Pengujian selanjutnya pada penelitian ini menggunakan Bahasa pemrograman C++, dengan struktur koding sebagai berikut :

```
#include <iostream.h>
#include <conio.h>
#include <stdio.h>
#include <math.h>
int a[100];
int max, min;
void maxmin(int i, int j) {
    int max1, min1, mid;
    if (i==j) {
        max=min=a[i];
    }
    else if (i==j-1) {
        if (a[i]>a[j]) {
```

```

max=a[i];
min=a[j]; }
else {
max=a[j];
min=a[i]; }
}
else {
mid=(i+j)/2;
maxmin(i, mid);
maxl=max;
minl=min;
maxmin(mid+1, j);
if (max<maxl)
max=maxl;
if(min>minl)
min=minl; }
}
void main() {
int i, num;
clrscr();
printf("\n\t\t>>Algoritma Djikstra");
printf("\n\t\t\tProgram Rute Maximum & Minimum
\n\n");
printf("\nMasukkan Jumlah Angka: ");
scanf("%d", &num);
printf("\nMasukkan Angkanya: \n");
for(i=0;i<num;i++) {
scanf("%d",&a[i]); }
max=a[0];
min=a[0];
maxmin(0, num-1);
printf("\nNilai Maximum Angka: %d\n",max);
printf("Nilai Minimum Angka: %d\n",min);
getch(); }

```

Dari koding diatas, maka hasilnya dapat ditampilkan sebagai berikut :

```

Lintasan Algoritma Terpendek (DIJKSTRA)
Masukkan nilai dari jalur antara simpul 1 dan 2: 2
Masukkan nilai dari jalur antara simpul 1 dan 3: 3
Masukkan nilai dari jalur antara simpul 1 dan 4: 5
Masukkan nilai dari jalur antara simpul 1 dan 5: 6

Masukkan nilai dari jalur antara simpul 2 dan 3: 5
Masukkan nilai dari jalur antara simpul 2 dan 4: 3
Masukkan nilai dari jalur antara simpul 2 dan 5: 2

Masukkan nilai dari jalur antara simpul 3 dan 4: 5
Masukkan nilai dari jalur antara simpul 3 dan 5: 6

Masukkan nilai dari jalur antara simpul 4 dan 5: 7

Masukkan asal simpul: 4
Masukkan target simpul: 5
ACEF
Jalur terpendek: 5_

```

Gambar 11. Tampilan Hasil Pengujian *Dijkstra* dengan C++

#### IV. KESIMPULAN

Dari hasil penelitian dengan menggunakan algoritma *dijkstra* dalam menentukan jalur terpendek untuk mencapai tempat wisata di Kota Palembang dengan mengambil sampel tempat wisata Pulau Kemaro dengan tujuan *node* akhirnya adalah jembatan Ampera dengan menggunakan taksi *Star Cab* untuk dapat mencapai tujuan tersebut sangat efektif dan efisien, sehingga dapat menghindari beberapa titik kemacetan dikota Palembang. Hal ini sangat bermanfaat bagi wisatawan atau pengunjung yang akan berkunjung ke Kota Palembang, selain cepat, menghemat waktu dan juga dapat menghemat biaya untuk menuju tempat wisata yang di tuju.

Dari hasil pembahasan algoritma *Dijkstra* dengan menggunakan Bahasa pemrograman C++ untuk menentukan jalur terpendek didapatkan *node* dengan hasil rute terpendek 5 km dengan jalur A, C, E, F. hal ini dapat mempersingkat waktu perjalanan dan menghemat biaya ongkos argo meter pada taksi *Star Cab*.

#### V. SARAN

1. Penelitian ini dengan menggunakan algoritma *dijkstra* dengan pengujian menggunakan Bahasa pemrograman C++, sehingga menghasilkan jarak dengan rute lebih pendek dari sebelumnya, dan diharapkan penelitian ini lebih bervariasi untuk penelitian selanjutnya.
2. Diharapkan penelitian selanjutnya lebih dilengkapi dengan GPS dan Google Maps.
3. Dan Diharapkan untuk penelitian selanjutnya dapat diimplementasikan berbasis android.

#### REFERENSI

- [1] Arifianto, Sofyan. 2012. *Sistem Aplikasi Penentuan Rute Terpendek Pada Jaringan Multi Moda Transportasi Umum Menggunakan Algoritma Dijkstra*. Hal 1-23. [http://eprints.undip.ac.id/36018/1/Sofyan\\_Arifianto.pdf](http://eprints.undip.ac.id/36018/1/Sofyan_Arifianto.pdf), diakses tanggal 19 Januari 2017.
- [2] Imam Sodikin. 2014. *Penentuan Rute Distribusi Produk Yang Optimal Dengan Memperhatikan Faktor Kecepatan Kendaraan Guna Meningkatkan Efisiensi Penggunaan BBM*. Prosiding Seminar Nasional Aplikasi Sains & Teknologi (SNAST). Hal 169-178. <http://repository.akprind.ac.id/sites/files/B169-178%20.%20Imam%20Sodikin.pdf>, diakses tanggal 26 Januari 2017.
- [3] Kriswanto, Y. Rudi, dkk. 2014. *Penentuan Jarak Terpendek Rute Transmisi dengan Algoritma Floyd-Warshall*. SEMINAR NASIONAL TEKNOLOGI INFORMASI & KOMUNIKASI TERAPAN 2. Hal 209 – 216. <http://eprints.dinus.ac.id/13730/1/Semantik->

- [047\\_Y\\_Rudi\\_Kriswanto\\_Arif\\_Aliyanto.pdf](#), diakses tanggal 26 Januari 2017.
- [4] Priatmoko, Shaga Bogas. Algoritma *Djiksra* Untuk Pencarian Jalur Terdekat Dan Rekomendasi Objek Pariwisata Di Pulau Bali. <https://www.scribd.com/doc/314068554/jurnal-13933-pdf>, diakses tanggal 29 Februari 2017.
- [5] Sholichin, Riyadhush, dkk. *Implementasi Algoritma Djiksra Dalam Pencarian Lintasan Terpendek Lokasi Rumah Sakit, Hotel Dan Terminal Kota Malang Berbasis Web*. <http://jurnal-online.um.ac.id/data/artikel/artikelC7BAA81E70E4BA519A5B64FEA7B04B3B.pdf>, diakses tanggal 2 Februari 2017.