Jurnal Sistem Informasi (JSI), VOL. 5, NO. 2, Oktober 2013, *Halaman 611-621*
ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

Implementasi Algoritma Dijkstra Dalam Aplikasi Untuk Menentukan Lintasan Terpendek Jalan Darat Antar Kota Di Sumatera Bagian Selatan

Fitria, Apri Triansyah
Department of Informatics Technique
The Informatics and Business Institute Darmajaya Bandar Lampung Indonesia 35142
Email: fitria_kenali@yahoo.com

Abstrak

Persoalan lintasan terpendek dapat diselesaikan dengan berbagai macam algoritma, salah satunya algoritma dijkstra. Algoritma ini menghitung bobot terkecil tiap-tiap titik sehingga tercapai nilai terkecil dari titik awal ke titik tujuan. Pada penelitian ini, algoritma dijkstra dipakai untuk menghitung jarak terdekat dari suatu kota ke kota lainnya pada sumatera bagian selatan. Hasil penelitian akan di wujudkan ke dalam bentuk perangkat lunak. Perangkat lunak ini akan di tempatkan pada fasilitas umum seperti terminal bus. Metode waterfall dipilih sebagai metode untuk mengembangkan perangkat lunak. Pengumpulan data dilakukan dengan metode : observasi dan studi pustaka. Sistem dirancang dalam beberapa tahapan yaitu pembuatan DFD, rancangan basis data, relasi antar tabel, rancangan flowchart dan rancangan interface.

Kata Kunci : Sumatera Bagian Selatan, Algoritma Dijkstra, Lintasan Terpendek, Metode Waterfall,, Bitmap.

1. Pendahuluan

Kita mengetahui bahwa untuk menuju ke suatu kota tujuan dapat ditempuh melalui beberapa lintasan. Dalam hal ini, kita akan menentukan kota-kota atau jalan manakah yang harus dilalui sehingga kita dapat mencari tempat tujuan dengan jarak terpendek. Dengan demikian lintasan terpendek dapat diartikan sebagai bobot minimal dari suatu lintasan, yaitu jumlah bobot dari seluruh busur yang membentuk lintasan. Dalam menentuan lintasan terpendek dapat diperoleh dengan beberapa algoritma matematika, antara lain algoritma *Dijkstra*, algoritma *Floyd-Warshall* dan algoritma *Bellman-Ford*. Algoritma yang akan di pakai dalam sistem ini adalah algoritma *Dijkstra*. Algoritma ini bertujuan untuk menemukan lintasan terpendek berdasarkan bobot terkecil dari satu titik ke titik lainnya. Misalkan titik mengambarkan kota, garis menggambarkan jalan dan bobot menggambarkan jarak, maka algoritma *Dijkstra* melakukan kalkulasi terhadap semua kemungkinan bobot terkecil dari setiap titik. Dengan kata lain algoritma ini menghitung lintasan berdasar jarak terpendek yang di tempuh di tiap-tiap kota.

1.1 Rumusan Masalah

Berdasarkan kondisi yang dijelaskan di atas, maka terdapat beberapa rumusan masalah yang akan dijawab dalam penelitian ini

- Bagaimana menentukan lintasan terpendek antar kota karena terdapat jalan yang bercabang-cabang.

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

- Bagaimana merancang dan menerapkan algoritma *Dijkstra* untuk melakukan kalkulasi terhadap semua kemungkinan bobot terkecil dari setiap titik.

1.2 Tujuan Penelitian

Penelitian ini bertujuan antara lain:

1. Memberikan solusi dalam pemilihan lintasan terpendek pada jalan darat antara kota-kota di sumatera bagian selatan meliputi provinsi Jambi, Sumatera Selatan, Bengkulu dan Lampung.
2. Mempercepat dalam mencari solusi lintasan terpendek pada jalan darat antara kota-kota di sumatera bagian selatan.
3. Memperoleh hasil yang akurat dan tepat sesuai dengan keadaan di lapangan.

1.3 Manfaat Penelitian

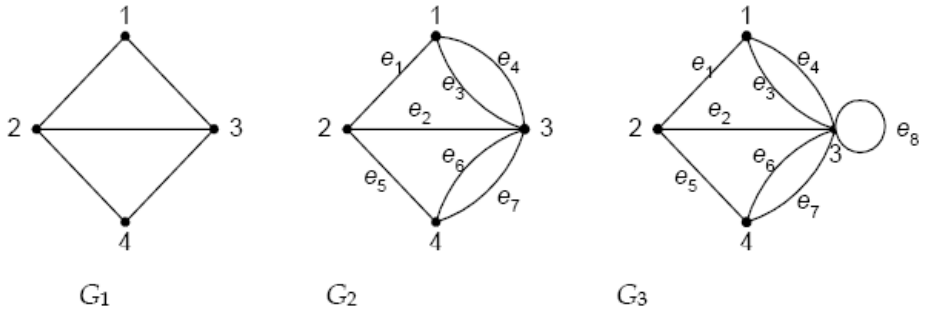
Dengan menerapkan alogaritma dijkstra untuk mencari lintasan terdekat dapat membantu para pengguna jalan, *traveling salesman*, perusahaan yang bergerak dibidang pariwisata dan angkutan antar provinsi, instansi pemerintah dan lain sebagainya terutama bagi yang membutuhkan informasi tentang lintasan terdekat.

2. TINJAUAN PUSTAKA

2.1 Teori Dasar Graf

Graf didefinisikan sebagai pasangan himpunan (V, E), ditulis dengan notasi G = (V, E), yang dalam hal ini V adalah himpunan tidak kosong dari simpul-simpul (*vertices* atau *node*) dan E adalah himpunan sisi (*edges* atau *arcs*) yang menghubungkan sepasang simpul [1]. Simpul pada graf dapat dinomori dengan huruf, seperti a, b, c...dst, dengan bilangan asli 1, 2, 3...dst, atau gabungan keduanya. Sedangkan sisi yang menghubungkan simpul u dengan simpul v dinyatakan dengan pasangan (u, v) atau dinyatakan dengan lambang e_1, e_2....e_n dengan kata lain, jika e adalah sisi yang menghubungkan simpul u dengan simpul v, maka e dapat ditulis sebagai e = (u, v). Secara geometri graf digambarkan sebagai sekumpulan noktah (simpul) di dalam bidang dwimatra yang dihubungkan dengan sekumpulan garis (sisi)

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index



Gambar 1. (G_1) graf sederhana, (G_2) multigraf, dan (G_3) multigraf

Gambar di atas memperlihatkan tiga buah graf, G_1, G_2 dan G_3 .
G_1 adalah graf dengan himpunan simpul V dan himpunan sisi E adalah:
V = {1, 2, 3, 4}
E = {(1, 2), (1, 3), (2, 3), (2, 4), (3, 4)}

G_2 adalah graf dengan himpunan simpul V dan himpunan sisi E adalah:
V = {1, 2, 3, 4}
E = {(1, 2), (2, 3), (1, 3), (1, 3), (2, 4), (3, 4), (3, 4)}
= {e_1, e_2, e_3, e_4, e_5, e_6, e_7}
G3adalah graf dengan himpunan simpul V dan himpunan sisi E adalah:
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}
Pada G_2, sisi e_3 = (1, 3) dan sisi e_4= (1, 3) dinamakan sisi-ganda (*multiple edges* atau *parallel edges*) karena kedua sisi ini menghubungi dua buah simpul yang sama, yaitu simpul 1 dan simpul 3. Pada G_3 , sisi e_8 = (3, 3) dinamakan gelang atau kalang (*loop*) karena ia berawal dan berakhir pada simpul yang sama.

2.2 Graf Berbobot (*Weighted Graph*)

Graf berbobot adalah graf yang setiap sisinya diberi sebuah harga (bobot) [1]. Bobot pada tiap sisi dapat berbeda-beda bergantung pada masalah yang dimodelkan dengan graf. Bobot dapat menyatakan jarak antara dua buah kota, biaya perjalanan antara dua buah kota, waktu tempuh pesan (*message*) dari sebuah simpul komunikasi ke simpul komunikasi lain (dalam jaringan komputer), ongkos produksi, dan sebagainya.

2.3 Lintasan Terpendek

Persoalan mencari lintasan terpendek di dalam graf merupakan salah satu persoalan optimasi. Graf yang digunakan dalam pencarian lintasan terpendek adalah graf berbobot (*weighted graph*), yaitu graf yang setiap sisinya diberikan suatu nilai atau bobot. Bobot pada sisi graf dapat menyatakan jarak antar kota, waktu pengiriman pesan, ongkos pembangunan, dan sebagainya. Asumsi yang kita gunakan di sini adalah bahwa semua

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

bobot bernilai positif. Kata terpendek berbeda-beda maknanya bergantung pada tipikal persoalan yang akan diselesaikan. Namun, secara umum terpendek berarti meminimisasi bobot pada suatu lintasan dalam graf [1].

2.4 Algoritma Dijkstra

Algoritma yang ditemukan oleh *Dijkstra* untuk mencari path terpendek merupakan algoritma yang lebih efisien dibandingkan algoritma *Warshall*, meskipun implementasinya juga lebih sukar. Misalkan G adalah graf berarah berlabel dengan titik-titik V(G) = {v_1, v_2,..., v_n} dan path terpendek yang dicari adalah dari v_1 ke v_n. Algoritma *Dijkstra* dimulai dari titik v_1. dalam iterasinya, algoritma akan mencari satu titik yang jumlah bobotnya dari titik 1 terkecil. Titik-titik yang terpiih dipisahkan dan titik-titik tersebut tidak diperhatikan lagi dalam iterasi berikutnya.

Misalkan:

V (G) = {v_1, v_2,.., v_n}

L	=	Himpunan titik-titik ε V (G) yang sudah terpilih dalam jalur path terpendek.
D(j)	=	Jumlah bobot path terkecil dari v_1 ke v_j.
w(i,j)	=	Bobot garis dari titik v_i ke v_j.
w*(1,j)	=	Jumlah bobot path terkecil dari v_1 ke vj

Secara formal, algoritma *Dijkstra* untuk mencari path terpendek adalah sebagai berikut:

1. L = { };
   V = {v_2, v_3,..,v_n}.
2. Untuk i = 2, ..., n, lakukan D(i) – w(1, i)
3. Selama $v_n \notin$ L lakukan:
   a. Pilih titik $v_k \in$ V - L dengan D(k) terkecil.
      L = L ∪ {v_k}
   b. Untuk setiap $v_j \in$ V - L lakukan:
      Jika D(j) > D(k) + W(k,j) maka ganti D(j) dengan D(k) + W(k,j)
4. Untuk setiap $v_j \in$ V, w*(1, j) = D(j)

Menurut algoritma di atas, path terpendek dari titik v_1 ke v_n adalah melalui titik-titik dalam L secara berurutan, dan jumlah bobot path terkecilnya adalah D(n).

Algoritma *Dijkstra* dinyatakan dalam *pseudo-code* berikut ini(Rinaldi Munir 2005 , hal. 414):

```
procedure Dijkstra (input m:matriks, a:simpul awal)
(
Mencari lintasan terpendek dari simpul awal a ke semua simpul lainnya
Masukan : matriks ketetanggaan (m) dari graf berbobot G dan simpul awal a
Keluaran : lintasan terpendek dari a ke semua simpul lainnya
)
Deklarasi
s1, s2, ..., sn :integer (tabel integer)
d1, d2, ..., dn :integer (tabel integer)
i, j, k: integer
Algoritma
( langkah 0 (Inisialisasi:)
for i ← 1 to n do
si ← 0
di ← mai
endfor
```

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

```
(langkah 1 :)
Sa← 1 (karena simpul a adalah simpul asal lintasan terpendek, jadi simpul a sudah pasti terpilih
dalam lintasan terpendek )
Da←oo (tidak ada lintasan terpendek dari simpul a ke a)
(langkah 2, 3, ..., n-1:)
For k← 2 to n-1 do
        J← simpul dengan sj = 0 dan dj minimal
        Sj ← 1 {simpul j sudah terpilih ke dalam lintasan terpendek}
        {perbaharui tabel d}
        For semua simpul I dengan si= 0 do
                If dj+mji <di then
                Di←dj+mji
                Endif
        Endfor
Endfor
)
```

2.4 Perangkat Lunak Pendukung

Borland Delphi 7

Kemampuan *Borland Delphi 7.0* secara umum adalah menyediakan komponen-komponen yang memungkinkan anda membuat program aplikasi yang sesuai dengan tampilan dan kerja *MS-Windows*, diperkuat dengan bahasa pemograman terstruktur yang sangat handal, yaitu struktur bahasa pemograman *object Pascal* yang sangat terkenal.

2.5 Corel Draw 12

Corel Draw 12 adalah sebuah aplikasi grafis berbasis *vector*. Format *vector* adalah gambar yang membentuk sejumlah objek garis dan objek kurva berdasarkan rumusan matematis. Format *vector* lebih banyak digunakan untuk membentuk objek buatan, seperti menggambar objek dua demensi, yang lebih ditekankan ke dalam pembuatan objek garis, lingkaran, polygon dan persegi panjang. Sedangkan untuk objek tiga dimensi, lebih ditekankan ke dalam pembuatan: bola, kubus dan tabung. Objek *vector*, banyak digunakan dalam pembuatan pengolahan teks dan logo [2].

2.5Microsoft Acces 2007

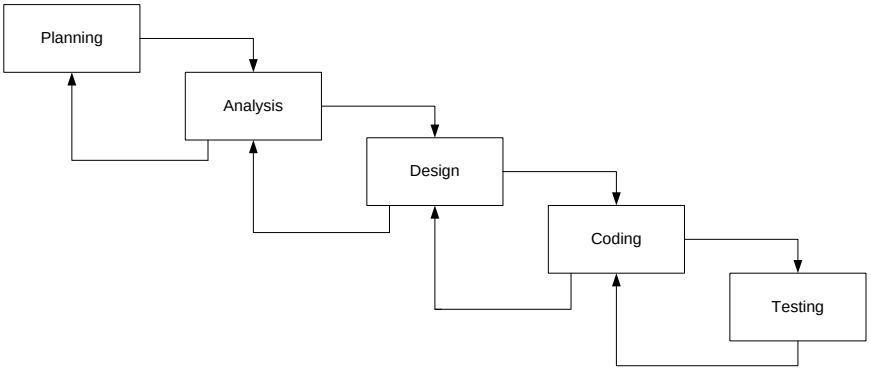
Merupakan salah program pengolah database yang cukup canggih dengan berbagai kemudahan yang ada seperti pengaturan data, pembuatan *form*, pembuatan laporan, menyaring data dan lain-lain. Maksud dari database itu sendiri adalah kumpulan arsip data berbentuk tabel yang saling berkaitan untuk menghasilkan informasi. Data sebagai masukan yang akan diolah menjadi informasi. Sedangkan informasi merupakan data yang telah diolah sesuai dengan kebutuhan. Informasi bagi satu pihak bias menjadi masukan bagi pihak lainnya.

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

3. Metode Penelitian

Metodologi yang digunakan Pada Implementasi Algoritma Dijkstra Dalam Aplikasi untuk menentukan Lintasan Terpendek Jalan Darat Antar Kota di Sumatera Bagian Selatan adalah Model Waterfall. Langkah awal dalam penelitian ini adalah mengumpulkan data, baik data primer maupun data sekunder. Hal ini dilakukan dengan menggunakan metode observasi, wawancara,dan studi dokumentasi/analisa arsip.Selanjutnya model waterfall ini mengusulkan sebuah pendekatan kepada perkembangan perangkat lunak yang sistematik dan sekuensial yang mulai pada tingkat dan kemajuan sistem pada sebuah Planning, analisis, desain, coding dan pengujian.

Untuk lebih jelasnya tahap-tahap dari paradigma waterfall dapat dilihat pada gambar dibawah ini



Gambar 2. Paradigma Waterfall (Classic Life Cycle)

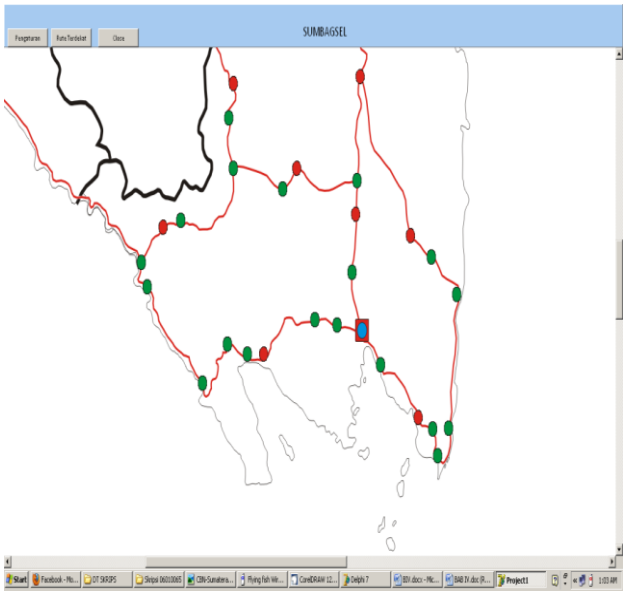
4. HASIL DAN PEMBAHASAN

Berikut ini merupakan menu awal ketika program dijalankan untuk pertama kali. Terdapat 3 buah menu pada program yang ditampilkan yaitu: *file, edit* dan *setting.kemudian tahapan berikutnya adalah login ganti password admin ,open view,atur tipe node,edit* bagian *distance edit distance, dan edit bagian layar yang akan dilakukan*

4.1 Halaman awal fgis

Berikut ini adalah tampilan untuk melihat peta yang sudah diubah/ditambahkan oleh admin. Tampilan ini dapat diakses oleh admin dan *user*. Untuk membukanya melalui menu open *view* lalu pilih *view* atau melalui menu *edit, map editor* kemudian lihat.

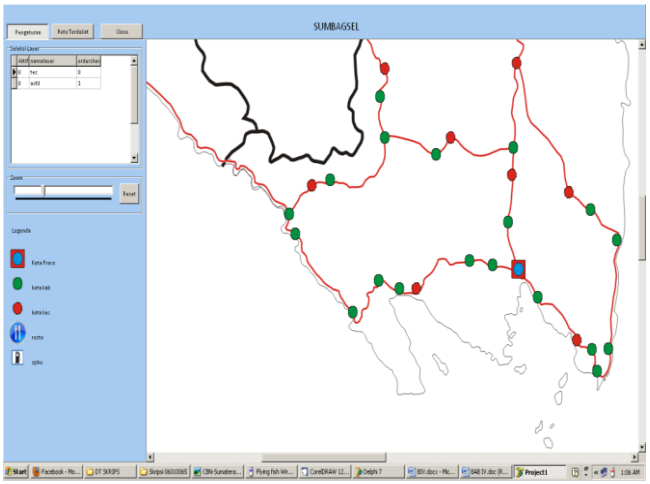
ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index



Gambar 3 Tampilan awal fgis/*map* .

4.2 Fgis bagian pengaturan

Berikut ini adalah tampilan untuk melakukan pengaturan dalam melihat peta yang ditampilkan. Tampilan ini dapat diakses oleh admin dan *user*. Untuk membukanya tekan pada tombol pengaturan pada tampilan awal fgis/*map*. Terdapat beberapa pengaturan antara lain pengaturan layer, kemudian pengaturan perbesaran/*zoom*, lalu legenda yang berisi keterangan pada *map*.



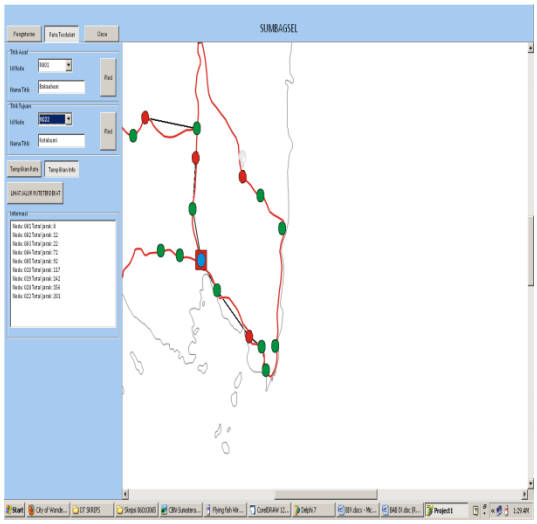
Gambar 4 Tampilan awal fgis/*map* bagian pengaturan.

4.3 Fgis bagian rute terdekat bagian pertama

Dala mencari rute terdekat terdapat 2 cara, ini merupakan cara pertama. Mula-mula *user* memilih titik awal dan titik tujuan pada *combobox*. Kemudian menekan tombol lihat jalur rute terdekat. Maka akan ditampilkan rute yang dilewati pada memo. Bila ingin melihat

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

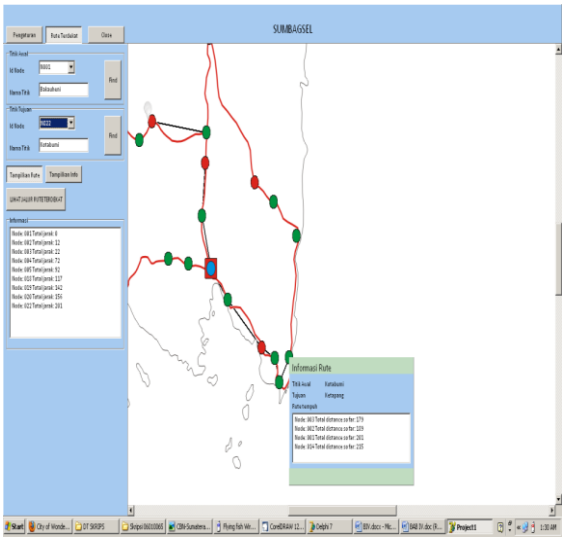
nama-nama titik/*node user* dapat menekan tombol tampilkan info lalu arahkan pada titik. Maka akan tampil nama *node*/titik tersebut.



Gambar 5 Fgis bagian rute terdekat bagian pertama.

4.4 Fgis bagian rute terdekat bagian kedua

Berikut ini adalah pencarian rute bagian kedua. User mula-mula menekan tombol tampilkan rute, lalu tentukan titik awal dengan mengklik *node* pada *map* dan arahkan ke titik tujuan. Secara otomatis akan tampil panel yang menunjukan *node* yang harus dilalui pada rute terdekat



Gambar 6 Fgis bagian rute terdekat bagian kedua.

4.5.1 Pengujian Program

4.5.2 Pengujian *blackbox* untuk *user*

Tabel pengujian program pada masukan bagian *user* ditunjukan oleh tabel 4.1 berikut ini.

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

Tabel 1. Pengujian pada program bagian user.

Input	Proses	Output	Hasil pengujian
Pemilihan menu *open view*	Openview1click	Menampilkan daftar *view* yang tersedia	Sesuai
Pemilihan *view* yang tersedia pada dbgrid	dobelklikDBGrid	Menampilkan *view* sesuai dengan yang dipilih pada fgis	Sesuai
Penekanan tombol pengaturan pada fgis	pengaturanButtonClick	Menampilkan panel pengaturan pada fgis	Sesuai
Klik kanan pada dbgrid layerkemudian pilih aktif	aktifkanClick	Merubah layer menjadi aktif (ditampilkan)	Sesuai
Klik kanan pada dbgrid layerkemudian pilih non aktif	nonaktifClick	Merubah layer menjadi non aktif (tidak ditampilkan)	Sesuai
Geser kekanan/kekiri pada zoomTrack	zoomTrackScroll	Memperbesar/memperkecil tampilan *map*	Sesuai
Tombol reset	resetButtonClick	Mengembalikan nilai *zoom* menjadi normal	Sesuai
Penekanan tombol rute terdekat pada fgis	dekatButtonClick	Menampilkan panel rute terdekat pada fgis	Sesuai
Pemilihan titik/*node* awal di *combobox*	Combobox1Change	Menampilkan nama titik/*node* pada edit1	Sesuai
Pemilihan titik/*node* tujuan di *combobox*	Combobox2Change	Menampilkan nama titik/*node* pada edit2	Sesuai
Penekanan tombol *find*	Find1Click	Mencari lokasi *node*/titik awal berdasar idnode di combobox1	Sesuai
Penekanan tombol *find*	Find2Click	Mencari lokasi *node*/titik awal berdasar idnode di combobox2	Sesuai
Penekanan tombol lihat jalur rute terdekat	dijkstraButtonClick	Menampilkan rute yang di lalui pada memo	Sesuai
Penekanan tombol tampilkan info	infoButtonClick	Menampilkan titik/*node* kota ketika mouse di arahkan ke titik/*node*	Sesuai
Penekanan pada masing-masing titik di fgis	nodeImageClick	Menekan titik pada fgis/*map* untuk dijadikan titik awal	Sesuai
Penekanan tombol tampilkan rute	ruteButtonClick	Menampilkan panel rute terdekat pada fgis ketika mouse diarahkan pada *node* tujuan (sebelumnya harus di tentukan titik awal)	Sesuai

4.5.2 Pengujian *blackbox* untuk admin

Tabel pengujian program pada masukan bagian admin ditunjukan oleh tabel 4.2 berikut ini.

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

Tabel 2. Pengujian pada program bagian admin.

Input	Proses	Output	Hasil pengujian
Pemilihan menu login admin	Loginadmin1Click	Menampilkan login admin	Sesuai
Penekanan tombol *ok* pada login admin	loginClick	Mengecek masukan *username* dan *password*, mengaktifkan menu admin	Sesuai
Penekanan tombol *close* pada login admin	closeButtonClick	Menutup login admin	Sesuai
Pemilihan menu *new view*	newView1Click	Menampilkan masukan tambah *view* baru	Sesuai
Tidak mengisi masukan pada tambah *view* lalu menekan simpan	simpanClick	Menampilkan event masukan masih ada yang kosong	Sesuai
Mengisi selain angka pada *edit* Bujur dan Lintang	EditBujurLintang keychange	Tidak menampilkan masukan selain angka	Sesuai
Penekanan pada tombol simpan di tambah *view*	simpanClick	Menyimpan data dan menampilkan pesan tersimpan	Sesuai
Pemilihan menu atur tipe ititik	Tipetitik1Click	Menampilkan atur tipe titik	Sesuai
Klik kanan pada dbgrid atur tipe dan pilih tambah	tambahClick	Menampilkan status tambah pada atur tipe	Sesuai
Klik kanan pada dbgrid atur tipe dan pilih *edit*	editClick	Menampilkan status *edit* pada atur tipe	Sesuai
Klik tombol *icon*	iconClik	Menampilkan explorer *file* gambar	Sesuai
Klik simpan pada atur tipe	simpanClick	Menyimpan data atur tipe	Sesuai
Klik kanan pada dbgrid atur tipe dan pilih hapus	hapusClick	Menghapus tipe *node*/titik	Sesuai
Klik pada submenu *map editor*	Mapeditor1click	Menampilkan dbgrid yang berisi daftar *view* yang akan diedit	Sesuai
Dobel Klik pada dbgrid yang berisi daftar *view*	dbgriddoubleClick	Menampilkan *view* pada fedit	Sesuai
Pilih tombol *node* pada fedit	nodeClick	Menampilkan panel *node*	Sesuai
Pilih tombol tambah pada panel *node*	tambahnodeClick	Menampilkan tambah panel	Sesuai
Klik sembarang pada *map* dalam kondisi tombol pada panel *node* terpilih	Image1Click	Menampilkan panel input *node* pada *map* di fedit	Sesuai

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

4.6 Instalasi Program

Setelah dilakukan pengujian terhadap program dan hasilnya cukup memuaskan, maka program siap untuk didistribusikan ke tempat yang membutuhkan informasi geografis mengenai rute terpendek. Program didistribusikan dalam bentuk *executable file* yang dikompresi sedemikian rupa. Hasil akhirnya adalah *file* setup.exe. Sebelum program diinstal, komputer yang digunakan sebagai sarana harus sudah terinstal *software* pendukung. Setelah semua lengkap, program diinstal di folder C:\Program Files\sigX\.

5. Kesimpulan

Berdasarkan latar belakang serta pembahasan-pembahasan pada bab sebelumnya, maka dapat disimpulkan bahwa :

a. Algoritma dijkstra dapat digunakan untuk mencari rute terpendek secara optimal.
b. Dengan menggunakan program ini dapat mempercepat dalam menentukan rute terpendek.
c. Program ini menawarkan beberapa kemudahan dalam menyusun peta secara dinamik sehingga apabila terdapat perubahan kondisi pada peta, program dapat menyesuaikan dengan kondisi baru.

DAFTAR PUSTAKA

[1] Munir, Rinaldi,2005. *Matematika Diskrit*. Bandung: Informatika Bandung.
http://www.scribd.com/doc/27745962/Microsoft-Access-2007, dikunjungi terakhir pada tanggal 22 agustus 2011, pukul 7.36 WIB.
[2] http://www.scribd.com/doc/16342804/Corel-Draw-12, dikunjungi terakhir pada tanggal 22 agustus 2011, pukul 7.37 WIB.