PERANCANGAN SISTEM INFORMASI GEOGRAFIS PENENTUAN JALUR JALAN OPTIMUM KODYA YOGYAKARTA MENGGUNAKAN ALGORITMA DIJKSTRA

Agus Qomaruddin Munir

Fakultas Sains & Teknologi, Universitas Respati Yogyakarta
Jl. Laksda Adisucipto Km. 6,3 Depok Sleman Yogyakarta 55281
E-mail: agusqmnr@yahoo.com

ABSTRAK

Sistem Informasi Geografis (SIG) adalah teknologi yang menjadi alat bantu dan sangat esensial untuk menyimpan, memanipulasi, menganalisis, dan menampilkan kembali kondisi-kondisi geografis. Terdapat 2 jenis data dalam Sistem Informasi Geografis, yaitu data spasial dan data non-spasial.Data spasial adalah data keruangan sebuah letak geografis, sedangkan data non-spasial menyatakan atribut dari letak geografis tersebut.

Salah satu problem dalam Sistem Informasi Geografis adalah pencarian jalur jalan. Penelitian ini bertujuan untuk merancang perangkat lunak untuk menyelesaikan problem tersebut yang akan diimplementasikan dengan studi kasus jaringan jalan di Kodya Yogyakarta. Rancangan berbentuk DFD, struktur data, dan algoritma yang berbasiskan Algoritma Dijkstra.

Hasil penelitian sudah dapat diimplementasikan meskipun diperlukan beberapa kajian untuk peningkatan efisiensi kinerja sistem.

Kata kunci: *SIG, jalur jalan optimum, Algoritma Dijkstra*

PENDAHULUAN

Perkembangan teknologi informasi sangat cepat seiring dengan kebutuhan akan informasi dan pertumbuhan tingkat kecerdasan manusia. Saat ini telah banyak sistem informasi yang digunakan untuk menunjang dan menyelesaikan suatu permasalahan yang biasanya timbul dalam suatu organisasi, perusahaan atau instansi pemerintahan. Sistem informasi diharapkan dapat meningkatkan kinerja dari suatu organisasi ataupun instansi agar lebih efektif dan efisien serta mudah dalam penerimaan informasi yang ingin disampaikan. Begitu juga dalam bidang Sistem Informasi Geografis (SIG) atau *Geographic Information System* (*GIS*) yaitu teknologi yang menjadi alat bantu dan sangat esensial untuk menyimpan, memanipulasi, menganalisis, dan menampilkan kembali kondisi-kondisi alam dengan bantuan data atribut dan keruangan.

Sistem Informasi Geografis (SIG) mempunyai kemampuan untuk dapat mengubah suatu sistem dari yang semula menggunakan konvensional yaitu sistem yang hanya dapat menampilkan data atribut saja menjadi sebuah sistem yang mempunyai basis grafis atau gambar berikut dengan data keruangan beserta atributnya.

Dalam perkembangannya Sistem Informasi Geografis dapat dijadikan sebagai alat bantu dalam mengambil keputusan, salah satu contohnya adalah untuk menempuh suatu perjalanan misalnya.

Untuk itu tujuan dari makalah akan merancang sebuah Sistem Informasi Geografis tentang ruas jalan- jalan di Kodya Yogyakarta beserta fasilitas-fasilitas umum yang ada. Sistem ini juga diharapkan dapat menentukan jalur jalan optimum dari 2 tempat berbeda, baik tempat itu berupa jalan maupun fasilitas umum.

LANDASAN TEORI

Konsep Dasar Sistem Informasi Geografis

SIG merupakan suatu sistem atau sekumpulan objek, ide yang saling berhubungan (inter-relasi) yang bertujuan dan bersasaran untuk menampilkan informasi geografis sehingga dapat mejadi suatu teknologi perangkat lunak sebagai alat bantu untuk pemasukkan, penyimpanan, manipulasi, analisis, dan menampilkan kembali kondisi-kondisi alam dengan bantuan data atribut dan keruangan. Pemahaman mengenai "dunia nyata" akan semakin baik jika proses-proses manipulasi dan presentasi data yang direlasikan dengan lokasi-lokasi geografis yang telah dimengerti [1][5][5].

Menurut beberapa ahli, sistem informasi geografis memiliki pengertian yang berbeda-beda. Berikut ini adalah definisi-definisi SIG [5][6]:

1. SIG adalah sistem komputer yang digunakan untuk memasukkan, menyimpan, memeriksa, mengintegrasikan, memanipulasi, menganalisa,

dan menampilkan data yang berhubungan dengan posisi di permukaan bumi.
2. SIG adalah kombinasi perangkat keras dan perangkat lunak komputer yang memungkinkan untuk mengelola, menganalisa, dan memetakan informasi spasial berikut data atributnya dengan akurasi kartografi.
3. SIG yang lengkap mencakup metodologi dan teknologi yang diperlukan yaitu data spasial, perangkat keras, perangkat lunak, dan struktur organisasi.
4. SIG adalah teknologi informasi yang dapat menganalisa, menyimpan, dan menampilkan baik data spasial maupun data non-spasial, yang mengkombinasikan kekuatan perangkat lunak basisdata relasional dan paket perangkat lunak CAD.

Didorong dengan perkembangan teknologi dan pasar SIG, paradigma perangkat lunak SIG telah mengalami perubahan beberapa kali [7], yaitu dari *GIS functional packages* hingga *integrated huge system*, dari *modular GIS* sampai *component GIS*, dari *desktop GIS* sampai *network-centric GIS*, dan dari *traditional client/server GIS* ke *distributed GIS*. Tiap-tiap perubahan tersebut menandai proses dalam sejarah perkembangan SIG.

Model Data Sistem Informasi Geografis

Model data yang akan digunakan dari bentuk dunia nyata harus diimplementasikan ke dalam basisdata. Data ini dimasukkan ke dalam komputer yang kemudian memanipulasi objek dasar yang memiliki atribut geometri (*entity* spasial/*entity* goegrafis)[5]. Secara umum persepsi manusia mengenai bentuk representasi *entity* spasial adalah konsep raster dan vektor. Dengan demikian data spasial direpresentasikan di dalam basisdata sebagai raster atau vektor. Dalam hal ini sering digunakan model data raster atau model data vektor.

Berikut merupakan model data Sistem Informasi Geografis[5]:

Data Raster

Model data raster memberikan informasi spasial apa yang terjadi di mana saja dalam bentuk gambaran yang digeneralisir. Dengan model ini, dunia nyata disajikan sebagai elemen matrik atau sel-sel *grid* yang homogen. Dengan model data raster, data geografi ditandai oleh nilai-nilai (bilangan) elemen matrik persegi panjang dari suatu objek. Dengan demikian, secara konseptual model data raster merupakan model data spasial yang paling sederhana.

Data Raster biasanya disimpan sebagai susunan dari nilai-nilai garis dengan *header* yang menyimpan metadata tentang susunan tersebut. Akurasi model data ini sangat bergantung pada resolusi atau ukuran pikselnya di permukaan bumi.

Data Vektor

Model data vektor menampilkan, menempatkan, dan menyimpan data spasial dengan menggunakan titik-titik, garis atau kurva, atau *polygon* beserta atribut-atributnya. Bentuk-bentuk dasar representasi data spasial ini, di dalam model data vektor, didefinisikan oleh sistem koordinat kartesian dua dimensi (x,y). Di dalam model data spasial *vector*, garis- garis atau kurva (busur atau *arcs*) merupakan sekumpulan titik-titik terurut yangdihubungkan. Sedangkan luasan atau *polygon* juga disimpan sebagai sekumpulan *list* (sekumpulan data atau objek yang saling terkait secara dinamis menggunakan *pointer*) titik-titik, tetapi dengan asumsi bahwa titik awal dan titik akhir *polygon* memiliki nilai koordinat yang sama (*polygon* tertutup sempurna).

Graph

Sebuah graph (*G*) dinyatakan sebagai pasangan tupel

$G = (V, E)$

dengan

V : himpunan berhingga verteks/node

E : himpunan berhingga edge.

Jika sebuah edge (v_1, v_2) terdapat di E maka v_1 dan v_2 terdapat di V. Sebuah graph disebut berbobot jika setiap edge (v_1, v_2) memiliki nilai yang disebut sebagai bobot. Sebuah edge dalam graph berbobot dinyatakan dalam bentuk (v_1, v_2, k) dengan

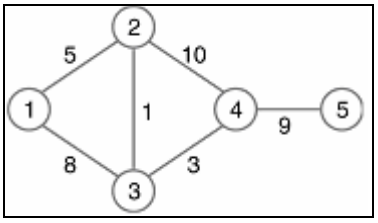
v_1 dan v_2 : verteks di V

k : bobot

Sebuah graph dapat dinyatakan dalam bentuk gambar dengan lingkaran menyatakan verteks dan busur yang menghubungkan lingkaran menyatakan sisi. Gambar 1 adalah contoh sebuah graph berbobot. Jika dinyatakan dalam bentuk G = (*V*, *E*) maka

$V = \{1, 2, 3, 4, 5\}$

$E = \{(1,2,5), (1,3,8), (2,3,1), (2,4,10), (3,4,3), (4,5,9)\}$



Gambar 1. Contoh graph berbobot

PEMBAHASAN

Analisis Kebutuhan

Kebutuhan data input atau masukan terdiri dari data spasial dan data non spasial. Data spasial dalam model vektor. Untuk kebutuhan data input atau masukan data spasial dari sistem informasi geografis ini adalah sebagai berikut:

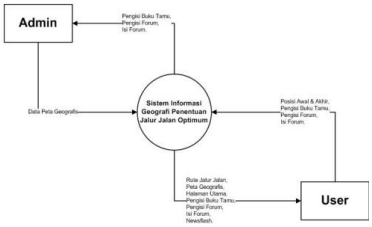
1. Data jalan, berupa data tentang kode jalan, nama jalan, layer jalan dan panjang jalan.
2. Data *network*, berupa data tentang panjang jalan, tipe jalan, kelas jalan, nama jalan dan layer *network.*
3. Data fasilitas umum, berupa data tentang kode fasilitas umum, nama fasilitas umum, tipe fasilitas umum, dan layer fasilitas umum.
4. Data sungai, berupa data tentang kode sungai, nama sungai, layer sungai, luas sungai dan keliling sungai.
5. Data *mainroad*, berupa data tentang kode *mainroad*, nama*mainroad*, luas *mainroad*, layer *mainroad* dan keliling *mainroad*.
6. Data *ringroad*, berupa data tentang kode *ringroad*, nama*ringroad*, luas *ringroad*, layer *ringroad* dan keliling *ringroad*.
7. Data rel KA, berupa data tentang kode *rel*, layer rel dan panjang rel.
8. Data wilayah, berupa data tentang kode wilayah, nama wilayah, luas wilayah, layer wilayah dan keliling wilayah.
9. Data POI (*point of interest*), berupa data tentang nama, fasilitas, tipe fasilitas dan layer fasilitas.

Sedangkan kebutuhan keluaran yang diharapkan adalah sebagai berikut:

1. Informasi tentang jalur jalan optimum/rute dari peta ruas jalan.
2. Informasi ruas jalan Kodya Yogyakarta.
3. Informasi letak fasilitas umum di Kodya Yogyakarta.
4. Informasi sungai di Kodya Yogyakarta.
5. Informasi mainroad di Kodya Yogyakarta.
6. Informasi ringroad di Yogyakarta.
7. Informasi wilayah di Propinsi Daerah Istimewa Yogyakarta.

Perancangan Sistem

Perancangan sistem menggunakan diagram aliran data (DFD, *Data Flow Diagram*). Rancangan sistem yang akan ditampilkan tidak hanya membahas rancangan sistem penentuan jalur optimum, tetapi juga rancangan sistem jika diimplementasikan dengan menggunakan web.



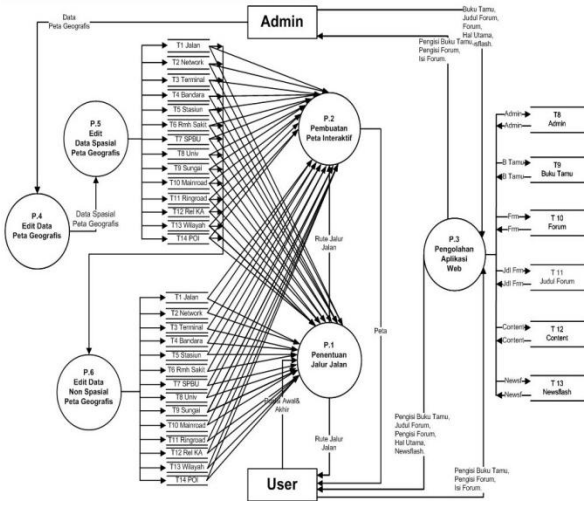
Gambar 2. DFD Level 0

Gambar 2 adalah rancangan DFD Level 0. Entitas luar Admin bertugas untuk mengelola sistem, sedangkan entitas luar User dapat menggunakan sistem untukmendapatkan informasi yang dibutuhkan. Data spasial berupa peta. Pada diagram, data spasial berupa Peta Geografis untuk setiap data spasial.

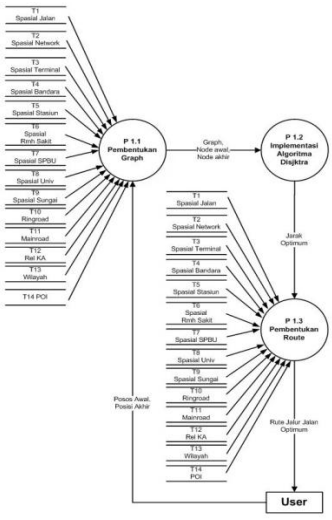
Informasi tentang jalur utama diberikan lewat data Rute Jalur Jalan. Untuk menentukan Rute Jalur Jalan, diperlukan data Posisi Awal dan Posisi Akhir. Posisi Awal dan Posisi Akhir dapat berupa jalan atau fasilitas umum.

Gambar 3 adalah DFD level 1 dari sistem. Proses utama dari sistem terdapat Proses 1 Penentuan Jalur Jalan. Penentuan Jalur Jalan membutuhkan data spasial, data non spasial, dan Posisi Awal serta Posisi Akhir dari jalur yang dicari.

Level 2 dari Proses Penentuan Jalur Jalan ditunjukkan oleh Gambar 4.



Gambar 3. DFD Level 1



Gambar 4. DFD Level 2 dari Proses Penentuan Jalur Jalan

Ada 3 subproses dari proses tersebut. Secara ringkas ketiga subproses itu adalah:

1. Proses Pembentukan Graph

Proses ini mengubah data spasial menjadi graph. Graph ini terbentuk dari semua data spasial jalan sehingga dapat dikatakan bahwa graph ini adalah representasi jaringan jalan. Verteks dari graph tersebut berupa persimpangan jalan. Sedangkan edge dari graph menunjukkan jalan yang menghubungkan setiap persimpangan, dengan bobot edge adalah jarak persimpangan.

Dengan model ini, sebuah jalan dapat membentuk lebih dari satu edge. Ini tergantung banyaknya persimpangan yang terdapat di jalan tersebut.

Selain persimpangan, maksimal 2 verteks dapat berupa sebuah lokasi fasilitas umum. Verteks ini adalah representasi dari posisi awal atau posisi akhir jika salah satu atau kedua posisi tersebut adalah fasilitas umum.

1. *Proses Implementasi Algoritma Dijkstra*

Proses ini menentukan jalur jalan optimum dari graph yang menyatakan jaringan jalan. Proses juga membutuhkan verteks awal dan verteks akhir yang menunjukkan posisi awal dan posisi akhir.

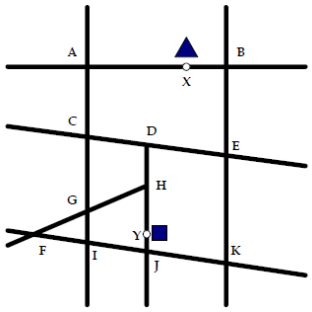
Hasil yang diberikan oleh proses ini berupa jalur dalam bentuk urutan verteks. Proses ini menggunakan algoritma Dijkstra [2][4].

2. *Proses Pembentukan Route.*

Proses ini membangkitkan gambar peta jaringan jalan dari data spasial sekaligus jalur jalan hasil penentuan Proses Implementasi Algoritma Disjktra.

Perancangan Struktur Data dan Algoritma

Berdasarkan perancangan sistem yang dibahas di subbab 3.2, diperlukan tipe data abstrak graph yang menyatakan jaringan jalan. Tipe data ini diperlukan untuk mempermudah pencarian jalur jalan optimum yang menghubungkan dua tempat, yang menggunakan algoritma Disjktra.

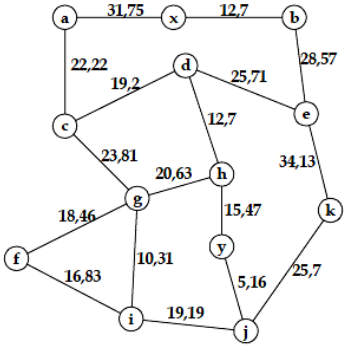


Gambar 5. Peta Jaringan Jalan

Misalkan diketahui sebuah peta jaringan jalan seperti yang ditunjukkan pada Gambar 5 dan akan dicari jalur jalan yang menghubungkan 2 lokasi yangdigambarkan dengan segitiga dan bujursangkar. Label huruf yang ada di peta hanya sebagai ilustrasi untuk memudahkan penjelasan. Label *as*/d *k* adalah persimpangan-persimpangan yang ada. Sedangkan label *x* dan *y* adalah posisi awal dan posisi akhir.

Graph berbobot yang dibentuk dari peta tersebut ditunjukkan pada Gambar 6. Bobot edge tidak harus menunjukkan jarak sebenarnya, tetapi jarak dalam peta juga bisa digunakan. Yang dipentingkan dari nilai bobot adalah menyatakan perbandingan jarak.

Misalkan diketahui sebuah graph $G=(V,E)$ dengan verteks awal dan verteks akhir adalah $x,y \in V$, S adalah himpunan verteks-verteks pembentuk jalur, $C(z)$ adalah label pada verteks z. $C(z) = [d, v]$ untuk label tetap atau $C(z) = [d,v]^*$ untuk label sementara, dengan d adalah jarak minimum z dari x dan v adalah verteks sebelumnya dari jalur x ke z yang minimum.



Gambar 6. Graph dari Jaringan Jalan

1. Inisialisasi verteks
   $S = \{x\}$
   $C(x) = [0, -]$

   $C(k)=[d,x]^*$, $k \in V - \{x\} \wedge (x,k,d) \in E$
   $[\infty,x]^*$ *otherwise*

2. Tentukan $m \in V$ --- Ssehingga $C(m) = [d,z]^*$ untuk sembarang z dan d minimum.
3. Ubah
   $C(m) = [d,z]$
   $S = S \cup \{m\}$
4. Jika $m=y$ maka ke langkah 7.
5. Untuk setiap $n \in V - S$, $(n,m,k) \in E$, dan $C(n) = [dn, v]^*$,

$$C(n) = \begin{cases} [d_n, v]^* & , d_n \le d + k \\ [d + k, m]^* & , otherwise \end{cases}$$

6. Kembali ke langkah 2.
7. Inisialisasi
   $P = y$
   $v = y$
8. Misal C(v) = [dv, w]
9. Jika $w=x$ maka berhenti.
10. Ubah
    $P = w \rightarrow P$
    $v = w$
11. Kembali ke langkah 8.

Misalkan Algoritma Dijkstra diterapkan untuk graph pada Gambar 6, sampai langkah ke-6, diperoleh nilai label untuk setiap verteks seperti tampak pada Tabel 1a dan Tabel 1b. Iterasi dihentikan pada saat diperoleh label tetap pada verteks *y*.
Dari Tabel 1a dan 1b, diperoleh jalur minimum dari x dan y adalah sebagai berikut:
x,b,e,d,h,y
dengan total bobot adalah *95,15*.

Jika jalur yang diperoleh tersebut dikembalikan ke peta maka jalan yang digambarkan dengan garis putus-putus adalah jalur yang dimaksud.

Tabel 1a. Algoritma Disjktra untuk graph pada Gambar 1.

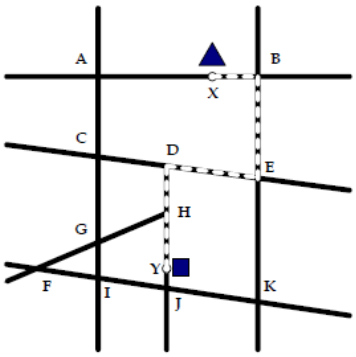
Iterasi	S	C(k)	m
0	{x}	C(x) : [0,-] C(a) : [31.75, x]* C(b) : [12.7, x]* C(c) : [∝, x]* C(d) : [∝, x]* C(e) : [∝, x]* C(f) : [∝, x]* C(g) : [∝, x]* C(h) : [∝, x]* C(i) : [∝, x]*	b
		C(j) : [∝, x]* C(k) : [∝, x]* C(y) : [∝, x]*	
1	{x, b}	C(x) : [0,-] C(a) : [31.75, x]* C(b) : [12.7, x] C© : [∝, x]* C(d) : [∝, x]* C(e) : [41.27, b]* C(f) : [∝, x]* C(g) : [∝, x]* C(h) : [∝, x]* C(i) : [∝, x]* C(j) : [∝, x]* C(k) : [∝, x]* C(y) : [∝, x]*	a
2	{x, b, a, e}	C(x) : [0,-] C(a) : [31.75, x] C(b) : [12.7, x] C© : [53.97, a]* C(d) : [∝, x]* C(e) : [41.27, b]* C(f) : [∝, x]* C(g) : [∝, x]* C(h) : [∝, x]* C(i) : [∝, x]* C(j) : [∝, x]* C(k) : [∝, x]* C(y) : [∝, x]*	e
3	{x, b, a, e}	C(x) : [0,-] C(a) : [31.75, x] C(b) : [12.7, x] C© : [53.97, a]* C(d) : [66.98, e]* C(e) : [41.27, b] C(f) : [∝, x]* C(g) : [∝, x]* C(h) : [∝, x]* C(i) : [∝, x]* C(j) : [∝, x]*	c
		C(k) : [75.4, e]* C(y) : [∝, x]*	
4	{x, b, a, e, c}	C(x) : [0,-] C(a) : [31.75, x] C(b) : [12.7, x] C© : [53.97, a] C(d) : [66.98, e]* C(e) : [41.27, b] C(f) : [∝, x]* C(g) : [77.78, c]* C(h) : [∝, x]* C(i) : [∝, x]* C(j) : [∝, x]* C(k) : [75.4, e]* C(y) : [∝, x]*	d
5	{x, b, a, e, c, d}	C(x) : [0,-] C(a) : [31.75, x] C(b) : [12.7, x] C© : [53.97, a] C(d) : [66.98, e] C(e) : [41.27, b] C(f) : [∝, x]* C(g) : [77.78, c]* C(h) : [79.68, d]* C(i) : [∝, x]* C(j) : [∝, x]* C(k) : [75.4, e]* C(y) : [∝, x]*	k

Tabel 2b. Algoritma Disjktra untuk graph pada Gambar 1.

Iterasi	S	C(k)	m
6	{x, b, a, e, c, d, k}	C(x) : [0,-] C(a) : [31.75, x] C(b) : [12.7, x] C(c) : [53.97, a] C(d) : [66.98, e] C(e) : [41.27, b] C(f) : [∝, x]* C(g) : [77.78, c]* C(h) : [79.68, d]* C(i) : [∝, x]* C(j) : [∝, x]* C(k) : [75.4, e] C(y) : [101, k]*	g
7	{x, b, a, e, c, d, k, g}	C(x) : [0,-] C(a) : [31.75, x] C(b) : [12.7, x] C(c) : [53.97, a] C(d) : [66.98, e] C(e) : [41.27, b] C(f) : [96.24, g]* C(g) : [77.78, c] C(h) : [79.68, d]* C(i) : [88.09, g]* C(j) : [∝, x]* C(k) : [75.4, e] C(y) : [101, k]*	h
8	{x, b, a, e, c, d, k, g, h}	C(x) : [0,-] C(a) : [31.75, x] C(b) : [12.7, x] C(c) : [53.97, a] C(d) : [66.98, e] C(e) : [41.27, b] C(f) : [96.24, g]*	i

		C(g) : [77.78, c] C(h) : [79.68, d] C(i): [88.09, g]* C(j) : [101, k]* C(k) : [75.4, e] C(y) : [95.15, h]*	
9	{x, b, a, e, c, d, k, g, h, i}	C(x) : [0,-] C(a) : [31.75, x] C(b) : [12.7, x] C(c) : [53.97, a] C(d) : [66.98, e] C(e) : [41.27, b] C(f) : [96.24, g]* C(g) : [77.78, c] C(h) : [79.68, d] C(i) : [88.09, g] C(j) : [101, k]* C(k) : [75.4, e] C(y) : [95.15, h]*	y
10	{x, b, a, e, c, d, k, g, h, I, y}	C(x) : [0,-] C(a) : [31.75, x] C(b) : [12.7, x] C(c) : [53.97, a] C(d) : [66.98, e] C(e) : [41.27, b] C(f) : [96.24, g]* C(g) : [77.78, c] C(h) : [79.68, d] C(i) : [88.09, g] C(j) : [101, k]* C(k) : [75.4, e] C(y) : [95.15, h]	

Untuk contoh tersebut, algoritma diterapkan terhadap graph tidak berarah, artinya jika $(v_1,v_2,k) \in E$ maka $(v_2,v_1,k) \in E$. Namun algoritma ini juga dapat diterapkan untuk graph berarah.
Dalam kasus jalur jalan, graph berarah ini diperlukan jika dalam peta terdapat jalan 1 jalur untuk sebagian jalan, dan jalan 2 jalur untuk jalan yang lain.



Gambar 7. Jalur jalan minimum dari x ke y.

KESIMPULAN

Rancangan yang ditampilkan dalam makalah ini diharapkan dapat diimplementasikan. Meskipun begitu perlu tinjauan lebih lanjut agar sistem dapat berjalan lebih efisien. Sebagai contoh, darirancangan diperoleh bahwa graph yang dibentuk akan besar jika peta semakin besar. Padahal ada kemungkinan bahwa sebagian verteks-verteks dari graph tidak akan digunakan dalam perhitungan karena terlalu jauh dari jalur yang akan dicari. Apalagi jika jalur yang dicari sebenarnya relatif cukup dekat dibandingkan dengan luasnya peta. Hal ini akan mengurangi efisiensi karena pembentukan graph juga memerlukan waktu.
Efisiensi juga menjadi masalah jika setiap dilakukan pencarian jalur, graph selalu dibentuk dari peta. Lebih baik jika graph dibentuk saat terdapat perubahan terhadap peta.

DAFTAR PUSTAKA

[1] Aronof, S. 1989. *Geographic Information System: A Management Perspective*. Ottawa: WDL Publications.
[2] Eklund, R., Kirkby, S., and Pollit, *A Dinamic Multi-source Dijkstra Algoritmh for Vehicle Routing*, http://www.kvocentral.com/kvopappers/jgis01.pdf, diakses tanggal 4 Oktober 2011.
[3] Juppenlatz, Morris. 1996. *Geographic Information System and Remote Sensing*. Sydney: McGraw Hill Book Company.
[4]Koch, R..*Dijsktra Algorithm*. http://www.nist.gov/dads/HTML/disjktraalgo.html, diakses tanggal 4 Oktober 2011.
[5] Prahasta, Eddy. 2002. *Konsep-Konsep Dasar Sistem Informasi Geografis*. Bandung: Informatika Bandung.
[6] Prahasta, Eddy. 2004. *Sistem Informasi Geografis Tools dan Plug-Ins*. Bandung: Informatika Bandung.
[7] Yuan, S. 2000. *Development of A Distributed Geoprocessing Service Model. University of Calagary*. Alberta.