Jurnal Ilmiah KURSOR
Menuju Solusi Teknologi Informasi
Vol. 4, No. 1, Januari 2008 ISSN 0216 - 0544

OPTIMASI *MULTI TRAVELLING SALESMAN PROBLEM (M-TSP)* PADA MOBIL PATROLI POLISI DENGAN ALGORITMA *HEURISTIC ASSIGNMENT* FISHER-JAIKUMAR DAN ALGORITMA A*

Santi Dwi Nurhumam[1]
Wayan Firdaus Mahmudy (wayanfm@brawijaya.ac.id) [2]

[1]Alumnus Program Studi Ilmu Komputer
Jurusan Matematika FMIPA Universitas Brawijaya

[2]Staf Pengajar Program Studi Ilmu Komputer
Jurusan Matematika FMIPA Universitas Brawijaya

ABSTRAK

Perjalanan n mobil patroli (n>1) ke sejumlah m titik rawan (m>n) termasuk dalam m-TSP (Multi Travelling Salesman Problem). Makalah ini memaparkan penyelesaikan permasalahan m-TSP dengan mengelompokkan terlebih dahulu m-1 (m titik tanpa titik start) titik rawan ke dalam n sub tur, dan satu titik rawan hanya boleh menjadi anggota sebuah sub tur. Pengelompokkan ini menggunakan metode heuristic assignment FisherJaikumar. Selanjutnya masing-masing sub tur direpresentasikan dalam bentuk graf dan dilakukan optimasi pencarian rute berdasarkan jarak pada masing-masing sub tur menggunakan algoritma A*, hasilnya berupa rute yang harus ditempuh oleh masing-masing mobil patroli. Evaluasi hasil dilakukan dengan cara membandingkan dengan solusi semua kemungkinan. Perbandingan dilakukan pada lima kali uji coba menggunakan tujuh, delapan, sembilan, sepuluh dan sebelas titik rawan. Pada perhitungan total jarak menggunakan metode heuristic assignment Fisher-Jaikumar dan algoritma A* menghasilkan galat rata-rata sebesar 20,43% dari jarak minimum sebenarnya. Sedangkan waktu pemrosesannya jauh lebih cepat untuk jumlah titik rawan lebih dari tujuh daripada dengan menghitung semua kemungkinan.

1. PENDAHULUAN

Transportasi merupakan salah satu permasalahan yang sering dihadapi dalam kehidupan sehari-hari. Salah satu contoh yaitu rute manakah yang memiliki biaya paling murah untuk dilalui seorang *salesman* yang harus mengunjungi sejumlah daerah, tiap daerah harus dikunjungi tepat satu kali kemudian kembali lagi ke tempat semula. Permasalahan tersebut dikenal sebagai *Travelling Salesman Problem (TSP)*. Jika terdapat lebih dari seorang *salesman* maka disebut *multi Travelling Salesman Problem (m-TSP)*.

Permasalahan *TSP* maupun *m-TSP* dimodelkan dalam bentuk graf lengkap. Pada kasus *m-TSP* graf lengkap tersebut memiliki n buah simpul yang menyatakan daerah-daerah yang harus dikunjungi oleh sejumlah m *salesman*. Bobot pada tiap garis pada graf tersebut menyatakan jarak tiap daerah. Secara matematis *m-TSP* bisa diformulasikan dalam persamaan [9]:

$$Z = \min\left\{\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}\right\} \quad (1)$$

dengan kendala

$$\sum_{i=1}^{n} x_{ij} = 1 \text{ untuk j = 1, 2, 3, ..., } n\text{-1} \quad (2)$$

$$\sum_{j=1}^{n} x_{ij} = 1 \text{ untuk i = 1, 2, 3, ..., } n\text{-1} \quad (3)$$

$$\sum_{i=1}^{n} x_{i1} = m \quad (4)$$

$$\sum_{j=1}^{n} x_{1j} = m \quad (5)$$

$x_{ij} = 1$, apabila ada perjalanan *salesman* dari simpul i menuju simpul j

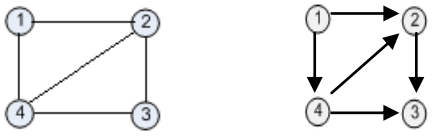
$x_{ij} = 0$, apabila tidak ada perjalanan *salesman* dari simpul i menuju simpul j

c_{ij}, menyatakan jarak dari simpul i menuju simpul j.

Persamaan (2) dan Persamaan(3) menjamin bahwa setiap simpul hanya dikunjungi sekali oleh *salesman.* Persamaan (4) dan Persamaan (5) menjamin bahwa sejumlah m *salesman* melakukan tur. Dalam tulisan ini simpul merupakan representasi dari kantor polisi dan pusat kerawanan yang harus dikunjungi sedangkan *salesman* merupakan representasi dari mobil patroli.

1.1. Graf

Sebuah graf G = (V,E) terdiri dari sebuah himpunan titik-titik V yang tidak boleh kosong serta himpunan garis-garis E (*edge*). Jika garis-garis tersebut diasumsikan sebagai pasangan berurutan dari titik-titik (v,w) maka disebut graf berarah; v disebut sebagai ekor dan w sebagai kepala dari garis (v,w). Jika garis-garis tersebut merupakan pasangan titik-titik yang tidak berurutan, tetap dilambangkan sebagai garis (v,w), namun disebut sebagai graf tidak berarah. Titik-titik tersebut biasa disebut sebagai node atau vertex. Sedangkan jumlah garis (edge) yang menghubungkan suatu node dengan node lain disebut derajat [1].



(a) (b)

Gambar 1 (a) Graf Tidak Berarah; (b) Graf Berarah.

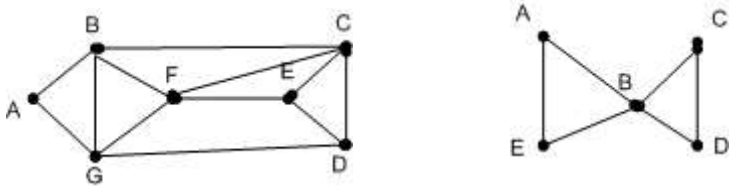
Gambar 1 (a) merupakan graf tidak berarah yang memiliki empat node yaitu node 1, 2, 3 dan 4. Node 1 memiliki derajat 2, node 2 memiliki derajat 3, node 3 memiliki derajat 2 dan node 4 memiliki derajat 4. Gambar 2 (b) merupakan graf berarah yang memiliki empat buah node yaitu node 1, 2, 3 dan 4. untuk jumlah derajat pada masing-masing node sama dengan Gambar 1 (a).

1.2. Graf Hamilton

Suatu graf disebut sebagai graf Hamilton apabila terdapat sirkuit dalam graf tersebut yang mengunjungi tiap titik tepat satu kali, kecuali titik awal yang sama dengan titik akhirnya. Apabila graf tersebut merupakan graf dengan lintasan terbuka (titik awalnya berbeda dengan titik akhirnya) maka disebut sebagai graf semi Hamilton [4].

Jika G merupakan graf Hamilton maka G mempunyai subgraf H dengan sifat-sifat sebagai berikut [10]:

1. H terhubung
2. H memuat semua titik G
3. H mempunyai jumlah garis yang sama dengan jumlah titiknya
4. setiap titik dalam H mempunya derajat 2.



(a) (b)

Gambar 2 (a) Graf Hamilton (b) Bukan Graf Hamilton

Misalkan graf pada Gambar 2 (a) adalah graf G, di dalamnya terdapat subgraf H yaitu ABCDEFGA. Subgraf tersebut telah memuat semua titik yang ada dalam graf G. Jumlah titiknya adalah tujuh dan jumlah garisnya adalah tujuh. Dengan kata lain jumlah titik dan jumlah garis dalam subgraf H adalah sama. Dalam subgraf H semua titik mempunyai derajat dua. Sehingga dapat disimpulkan bahwa graf G adalah graf Hamilton.

Misalkan Gambar 2 (b) adalah graf G. Bisa dibentuk subgraf H yang memuat semua titiknya yaitu ABDCBEA atau AEBCDBA ataupun yang lain, namun tetap saja B berderajat empat. Sehingga subgraf H tidak memenuhi syarat yang keempat.

Dalam suatu kasus yang membutuhkan graf Hamilton, mungkin saja terjadi masukan yang ada berupa graf terbuka yang berarti bukan merupakan graf Hamilton. Permasalahan seperti ini termasuk dalam *Graphical Travelling Salesman Problem* (GTSP). Permasalahan ini bisa diselesaikan dengan cara menambahkan busur pada pasangan simpul yang belum bertetangga.

1.3. Generalized Assignment Problem (GAP)

Untuk menyelesaikan m-TSP mula-mula harus dilakukan pembagian sejumlah titik rawan kepada sejumlah salesman. Permasalahan tersebut termasuk dalam *Generalized Assignment Problem* (GAP) [5]. Total biaya yang dikenakan bisa berubah tergantung pada titik mana saja yang harus dikunjungi oleh seorang salesman (*agen-task assignment*). Dalam GAP salesman direpresentasikan sebagai agen dan titik rawan sebagai pekerjaan yang harus dikerjakan oleh agen. Dalam permasalahan ini dibutuhkan tepat seorang agen untuk menyelesaikan masing-masing pekerjaan sehingga bisa diperoleh total biaya yang minimum. Dengan kata lain jika diberikan himpunan N yang berisi sejumlah n pekerjaan dan himpunan M yang berisi sejumlah m agen (tiap pekerjaan harus diselesaikan oleh tepat seorang agen). Formula dari permasalahan tersebut ditunjukkan pada Persamaan 6.

$$\max \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \tag{6}$$

dengan kendala

$$\sum_{j=1}^{m} x_{ij} = 1 \qquad i = 1..., n \tag{7}$$

$$\sum_{i=1}^{n} a_{ij}x_{ij} \le b_j \quad j = 1..., m \tag{8}$$

$$x_{ij} \in \{0,1\} \qquad i = 1..., n, \quad j = 1...,m$$

$i \in N$ dan $j \in M$, x_{ij} adalah variabel yang berarti apakah pekerjaan i dikerjakan oleh agen j atau tidak, c_{ij} adalah biaya yang dibutuhkan agen j untuk mengerjakan pekerjaan i,. a_{ij} adalah jumlah sumber (*resource*) yang dibutuhkan untuk oleh pekerjaan i ketika diselesaikan oleh agen j, sedangkan b_j adalah jumlah sumber yang dimiliki oleh agen j [2]. Salah satu metode yang bisa digunakan untuk menyelesaikan GAP adalah algoritma heuristik Fisher-Jaikumar.

1.4. Algoritma Heuristic Assignment Fisher-Jaikumar

Algoritma Fisher-Jaikumar adalah sebuah algoritma heuristik yang dikembangkan oleh Fisher dan Jaikumar pada tahun 1981. Berikut ini langkah-langkah untuk menyelesaikan kasus m-TSP dengan algoritma Fisher-Jaikumar [2] :

1. Diberikan masukan berupa matrik jarak dari tiap-tiap kota serta jumlah *salesman*.
2. Dipilih salah satu kota sebagai titik awal (*start node*).
3. Dilakukan pengecekan apakah jumlah kota kurang dari atau sama dengan jumlah *salesman* kurang satu, jika tidak, jumlah *salesman* diubah menjadi sama dengan jumlah kota kurang satu, kemudian dilakukan pemilihan pusat *cluster* yaitu sesuai dengan jumlah kota kurang satu. Jika jumlah kota sudah kurang dari atau sama dengan jumlah *salesman*, maka langsung dilakukan pemilihan pusat *cluster*.
4. Memilih pusat *cluster* dengan cara memilih simpul tetangga *start node* mulai dari jarak terkecil sebanyak jumlah *salesman*.
5. Diperoleh pusat *cluster* sebanyak jumlah *salesman*.
6. Untuk setiap kota yang belum dikunjungi (bukan termasuk *start node* maupun pusat *cluster*), kota tersebut ditentukan untuk masuk dalam sub tur (*cluster*) yang mana, dengan cara dicari pusat *cluster* yang bertetangga paling dekat dengan kota tersebut. Jika terdapat lebih dari satu pusat *cluster* memiliki jarak terkecil yang sama maka prioritaskan pada *cluster* dengan jumlah titik rawan terkecil. Ulangi langkah

tersebut sampai semua kota telah dikunjungi.
7. Dihasilkan sub tur sebanyak jumlah *salesman*.
8. Algoritma ini memberikan *output* berupa matrik jarak kota pada masing-masing sub tur.

1.5. Algoritma A*

Algoritma A* pertama kali diperkenalkan pada tahun 1968 oleh Peter Hart, Nils Nilsson dan Bertram Raphael. Dalam tulisan mereka, algoritma ini disebut sebagai algoritma A. Algoritma A merupakan salah satu algoritma pencarian graf, yaitu menemukan jalur terpendek dari titik awal sampai ke titik tujuan dalam suatu graf. Algoritma ini menggunakan perkiraan heuristic h(x) yang menyusun tiap titik x berdasarkan perkiraan rute terbaik yang melalui titik tersebut. Kemudian titik yang dikunjungi adalah titik dengan perkiraan terbaik tersebut [3][6].

Algoritma A bekerja dengan menyimpan sebuah antrian prioritas dari garis-garis yang melalui graf tersebut dimulai dari titik awal. Misalkan dari titik awal (start node) ke titik A terdapat sebuah garis, prioritas pada garis x berdasarkan Persamaan 9

$$f(x) = g(x) + h(x) \quad (9)$$

g(x) merupakan biaya yang dimiliki oleh garis yang menghubungkan dari titik awal menuju ke titik x. Sedangkan h(x) adalah biaya minimum yang sebenarnya untuk mencapai titik tujuan dari titik x. Nilai f(x) yang paling rendah mendapat prioritas paling tinggi dalam antrian tersebut.

Algoritma A ini kemudian dikembangkan sehingga menjadi algoritma A*. Jika dalam algoritma A digunakan nilai h(x) maka dalam algoritma A* digunakan nilai h'(x) yang merupakan perkiraan biaya untuk sampai pada suatu tujuan dari titik x. Dengan syarat $h'(n) \leq h(x)$. Untuk evaluasinya algoritma ini menggunakan fungsi seperti yang terlihat pada Persamaan 10

$$f'(x) = g(x) + h'(x) \quad (10)$$

Beberapa istilah yang digunakan dalam algoritma ini antara lain *open list*, *closed list*, *start node*, *goal node* dan *parent*. Open list diumpamakan seperti daftar belanjaan. Dalam algoritma ini open list berisi daftar titik (node) yang memiliki kemungkinan untuk dikunjungi berikutnya. Jika dari open list sudah ditemukan titik yang dipilih sebagai jalur berikutnya maka bisa dimasukkan dalam closed list. Dengan closed list ini bisa diketahui node-node yang telah diperiksa. Pada kasus perjalanan mobil patroli ini start node adalah titik rawan awal yang digunakan sebagai titik berangkat semua mobil patroli. Goal node adalah node tujuan untuk masing-masing mobil patroli. Parent adalah node yang dikunjungi tepat sebelum sebuah node. Berikut ini pseudo-code algoritma A* [5]:

```
function A*(start,goal)
var closed := the empty set
var q := make_queue(path(start))
while q is not empty
    var p := remove_first(q)
    var x := the last node of p
    if x in closed
        continue
    if x = goal
        return p
    add x to closed
    foreach y in successors(p)
        enqueue(q, y)
return failure
```

1.6. Sistematika Perjalanan Mobil Patroli Keamanan Polisi

Pada tingkat kota atau Polresta, patroli dibagi menjadi dua bagian yaitu patroli terbuka dan patroli tertutup. Patroli terbuka adalah patroli yang dilakukan oleh anggota kepolisian bagian lalu lintas dan SABARA (Samapta Bayangkara). Sedangkan pada patroli tertutup yang dilaksanakan oleh bagian Intel dan Serse, lebih ditujukan untuk pengawasan orang asing, teroris, atau adanya kegiatan yang dicurigai mengarah pada tindak kriminal. Jadi

pelaksanaan patroli tertutup ini hanya sewaktu-waktu jika ada perintah untuk melakukan pengawasan terhadap sesuatu atau seseorang yang dianggap mencurigakan.

Tulisan ini menggunakan kasus pelaksanaan patroli keamanan terbuka di wilayah kota yang dilakukan oleh SABARA. Dalam teorinya, semua patroli wajib dilakukan secara terus-menerus baik pagi, siang maupun malam. Namun mengingat adanya keterbatasan biaya operasional untuk kendaraan dan keterbatasan manusia, maka patroli hanya dilakukan sesekali. SABARA biasanya melakukan patroli hanya sekali dalam sehari melalui rute yang telah ditentukan. Jadi dalam sekali patroli, tidak semua tempat-tempat vital bisa dikunjungi. Hal ini memungkinkan terjadinya tindak kriminal pada tempat-tempat tidak dikunjungi karena tidak adanya pengawasan yang lebih intensif dari pihak kepolisian, hanya ada pengawasan dari pihak keamanan yang berjaga di tempat tersebut. Karena kelemahan-kelemahan tersebut, maka pelaksanaan patroli keamanan belum bisa mencapai hasil yang maksimal.

1.7. Pencarian Rute Minimum dengan Menghitung Semua Kemungkinan

Untuk mengetahui nilai pencarian rute yang paling minimum bisa dilakukan dengan cara menghitung semua kemungkinan yang ada sebagai berikut:

1. Dilakukan pendataan rute-rute yang mungkin untuk dilalui.
2. Kemudian dilakukan perhitungan jarak yang diperoleh dari masing-masing kemungkinan rute yang ada.
3. Selanjutnya jarak dari masing-masing kemungkinan rute yang telah dihitung tersebut dibandingkan untuk diambil jarak yang paling minimum.
4. Jarak paling minimum inilah yang menjadi hasil pencarian rute minimum dengan menghitung semua kemungkinan.

1.8. Analisis Galat

Galat (*error*) adalah selisih antara solusi sebenarnya (*exact solution*) dengan solusi hampiran (*approximation*). Galat berasosiasi dengan seberapa dekat solusi hampiran terhadap solusi sebenarnya. Semakin kecil galatnya, semakin teliti solusi numerik yang didapatkan [7]. Selisih (galat) antara nilai sebenarnya dengan nilai hampiran ditunjukkan dalam Persamaan 11.

$$\varepsilon = a - \hat{a} \tag{11}$$

Keterangan :
ε : galat
a : nilai sebenarnya
$\hat{a}$: nilai hampiran

Karena nilai positif atau negatif tidak dipertimbangkan, maka galat didefinisikan sebagai suatu nilai mutlak seperti ditunjukkan dalam Persamaan 12.

$$|\varepsilon| = |a - \hat{a}| \tag{12}$$

Dari Persamaan 11 dan Persamaan 12 tidak terdapat informasi seberapa besar galat yang terjadi pada nilai hampiran dibandingkan dengan nilai sebenarnya. Sehingga bisa menimbulkan nilai galat yang rancu untuk kasus yang berbeda. Untuk menghindari kerancuan interpretasi nilai galat, maka nilai galat tersebut harus dinormalkan terhadap nilai sebenarnya dan disebut sebagai galat relatif, seperti ditunjukkan dalam Persamaan 13 dan Persamaan 14.

$$\varepsilon_R = \frac{\varepsilon}{a} \tag{13}$$

Atau dalam bentuk presentase

$$\varepsilon_R = \frac{\varepsilon}{a} \text{ x } 100\% \tag{14}$$

2. METODE

Tahapan pencarian rute optimal dilakukan perangkat lunak dengan langkah-langkah beikut:

1. Masukan pada sistem diperoleh dari pengguna, yaitu dengan cara pengguna memilih peta apa yang ingin dianalisanya, jumlah daerah rawan serta jumlah mobil yang ingin dipergunakan.
2. Sistem yang menerima masukan tersebut kemudian akan mengambil *database* yang dimiliki oleh peta tersebut yaitu data matrik jarak dan koordinat titik kerawanan yang dimiliki oleh peta tersebut.
3. Dari data tersebut kemudian dilakukan pengolahan dalam sistem dengan menggunakan algoritma *heuristic* Fisher-Jaikumar sehingga terbentuklah beberapa sub tur.
4. Pada masing-masing sub tur dilakukan optimasi pencarian rute terpendek dengan algoritma A*.
5. Hasil akhir dari pengolahan tersebut berupa total jarak yang ditempuh serta rute terpendek yang dihasilkan. Hasil tersebut diolah dulu menjadi graf baru kemudian hasilnya dikembalikan pada pengguna.

Untuk mengetahui optimal atau tidaknya program aplikasi ini maka dilakukan lima kali uji coba pada sistem menggunakan data simulasi masing-masing sebanyak tujuh, delapan, sembilan, sepuluh dan sebelas titik rawan dengan asumsi titik (1) adalah kantor polisi dan titik-titik rawan yang lainnya berupa pusat-pusat keramaian yang menjadi prioritas untuk dilakukan patroli, seperti hotel, tempat industri, pusat perbelanjaan dan sebagainya. Sedangkan jumlah mobil yang digunakan sebanyak dua buah mobil patroli. Pada masing-masing uji coba diberikan koordinat titik rawan yang berbeda-beda.

Uji coba dilakukan dalam dua tahap, yang pertama yaitu dilakukan pencarian rute minimum pada data sampel menggunakan program aplikasi dengan metode *heuristic assignment* dan algoritma A*, selanjutnya dilakukan pencarian rute minimum dengan cara menghitung semua kemungkinan rute pada data sampel.

3. HASIL DAN PEMBAHASAN

Untuk mengetahui keoptimalan hasil perhitungan metode heuristic assignment Fisher-Jaikumar dan Algoritma A*, berikut ini diberikan tabel perbandingannya dengan hasil perhitungan pada semua kemungkinan jalur untuk data sampel.

Tabel 1. Perbandingan Hasil Perhitungan

Uji Ke	Jumlah Titik Rawan	Total Jarak (piksel)			Waktu Proses (milidetik)	
		Metode Heuristic Assignment dan Algoritma A*	Semua Kemungkinan Jalur	Galat (%)	Metode Heuristic Assignment dan Algoritma A*	Semua Kemungkinan Jalur
1	7	2327	1624	43,29	0	15
2	8	1221	1149	6,27	15	31
3	9	2178	1599	36,21	16	281
4	10	1409	1365	3,22	14	5905
5	11	2623	2318	13,16	16	151093
Rata-Rata				20,43		

Dari hasil pengujian pada Tabel 1, bisa diketahui bahwa selisih jarak yang dihasilkan oleh algoritma *heuristic assignment* Fisher-Jaikumar dan algoritma A* dengan jarak minimum sebenarnya bervariasi. Algoritma A* merupakan metode heuristik sehingga total jarak minimum yang diperoleh hanyalah pendekatan dari jarak minimum yang sebenarnya. Tetapi secara umum dengan bertambahnya titik rawan, selisih total jarak minimum (ditunjukkan oleh nilai galat) yang dihasilkan cenderung semakin menurun.

Untuk waktu pelaksanaan proses, metode heuristic assignment Fisher-Jaikumar dan algoritma A* hanya mengalami penambahan waktu yang sedikit pada tiap penambahan jumlah titik rawan. Sedangkan pencarian rute dengan menghitung semua kemungkinan yang ada memerlukan waktu hampir sama dengan algoritma heuristic assignment Fisher-Jaikumar dan algoritma A* pada saat jumlah titik rawan kurang dari delapan. Namun untuk jumlah titik rawan lebih dari delapan, waktu yang dibutuhkan untuk melakukan proses meningkat drastis. Hal ini disebabkan penambahan jumlah kemungkinan rute yang harus dihitung akan semakin besar tiap kali dilakukan penambahan jumlah titik rawan. Penambahan jumlah kemungkinan rute yang harus dihitung di tunjukkan pada persamaan 15.

```
Jumlah kemungkinan rute = ((n-1)!) x Jumlah Kombinasi
```
(15)

n adalah jumlah titik rawan sedangkan yang dimaksud jumlah kombinasi adalah jumlah kemungkinan pengelompokan titik-titik rawan ke dalam sejumlah mobil patroli. Misalkan untuk titik rawan berjumlah tujuh dengan dua buah mobil patroli, memiliki jumlah kombinasi sebanyak tiga yaitu kombinasi pertama dua titik rawan untuk mobil pertama dan enam titik rawan untuk mobil kedua. Kombinasi kedua terdiri dari tiga titik rawan untuk mobil pertama dan lima titik rawan untuk mobil kedua. Dan kombinasi yang terakhir terdiri dari empat titik rawan untuk mobil pertama dan empat titik rawan juga untuk mobil kedua. Dengan asumsi titik rawan (1) adalah kantor polisi sehingga digunakan sebagai start node dan goal node untuk semua mobil patroli.

4. KESIMPULAN

- Pencarian rute menggunakan algoritma heuristic assignment Fisher-Jaikumar dan Algoritma A* menghasilkan nilai jarak yang mendekati optimum dengan tingkat kesalahan rata-rata sebesar 20,43%.
- Waktu yang dibutuhkan untuk menyelesaikan suatu pencarian rute dengan metode heuristic assignment Fisher-Jaikumar dan algoritma A* lebih cepat daripada menghitung semua kemungkinan rute.
- Penggunaan algoritma heuristic assignment Fisher-Jaikumar dan algoritma A* lebih cocok untuk jumlah data yang besar. Sedangkan untuk data yang berukuran kecil lebih baik dengan menghitung semua kemungkinan rute yang ada karena hasil yang diperoleh pasti optimal.

DAFTAR PUSTAKA

[1] Aho, Alfred V et al. 1974. *The Design and Analysis of Computer Algorithm*. America: Addison-Wesley Publishing Company.

[2] Alfandari, Laurent et al. 2001. *A Two-Phase Path-Relinking Algorithm for the Generalized Assignment Problem.* 4 th Metaheuristic International Conference (MIC). Porto, Portugal.

[3] Anonymous. *A* search algorithm*, diakses pada tanggal 3 Pebruari 2007 dari http://en.wikipedia.org/wiki/A-star_algorithm.htm

[4] Anonymous. *Hamiltonian path*, diakses pada tanggal 6 Maret 2007 dari http://en.wikipedia.org/wiki/hamiltonian_path.htm.

[5] Koskosidis, Yiannis A & Powell, Warren B. 1992. *Clustering Algorithms for Consolidation of Customer Orders into Vehicle Shipments*. Transpn. Res. –B, 26B(5): 365-379.

[6] Lester, Patrick. 2005. *A* Pathfinding for Beginners*. Diakses pada tanggal 4 Pebruari 2007 dari http://www.policyalmanac.org/games/aStarTutorial.htm.

[7] Munir, Rinaldi. 2003. *Metode Numerik.* Bandung: Informatika Bandung.

[8] Munir, Rinaldi. 2004. 1. *Algoritma Brute Force (lanjutan) 2. Heuristik. Bahan Kuliah-2 Strategi Algoritmik*. Departemen Teknik Informatika, Institut Teknologi Bandung, Bandung.

[9] Setiyono, Budi. 2002. *Pembuatan Perangkat Lunak Penyelesaian Multi Travelling Salesman Problem (m-TSP). KAPPA*, 3(2): 55-65.

[10] Siang, Jong Jek. 2002. Matematika Diskrit dan Aplikasinya pada Ilmu Komputer. Yogyakarta: Andi.