Penyelesaian Masalah *Vehicle Routing and Scheduling* Dalam Dunia Nyata Berdasarkan Algoritma Genetika

Emiliana Dewi Aryani[1], Fangyan Dong[2], Kewei Chen[3], Yasufumi Takama[2], dan Kaoru Hirota[2]

[1] Fakultas Ilmu Komputer, Universitas Indonesia, emil96@puspa.cs.ui.ac.id
[2]Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology, Japan
{fydong, takama, hirota}@hrt.dis.titech.ac.jp
[3]J and F Co. Ltd, VZW03122@nifty.ne.jp

Abstrak - **Berdasarkan konsep *Vehicle Routing, Scheduling, and Dispatching Problem with Single Depot (VRSP/SD)* yang telah diajukan oleh Chen, suatu metode untuk memecahkan persoalan *Vehicle Routing and Scheduling* dengan satu depot (*Vehicle Routing and Scheduling Problem with Single Depot* (VRSP/SD)) berdasarkan metode algoritma genetika diajukan pada makalah ini. Matriks dua dimensi digunakan sebagai representasi jadwal (*schedule*). Operasi *crossover* dilakukan melalui operasi pemindahan atau operasi pertukaran sekelompok pelanggan (*user/customer*). Percobaan dilakukan berdasarkan data pengiriman makanan harian dari satu depot ke 46 toko di daerah Saitama, Jepang. Perbandingan hasil eksperimen dengan metode sebelumnya yang menggunakan metode *Simulated Annealing* menunjukkan bahwa metode yang diajukan menghasilkan hasil yang sama baiknya dengan metode terdahulu.**

Kata kunci : vehichle routing, scheduling, crossover, simulated annealing

Makalah diterima [10 April 2001]. Revisi akhir [28 April 2001]

1. PENDAHULUAN

Sekarang ini, masalah transportasi telah menjadi salah satu titik penting dalam meningkatkan produktivitas. Pengurangan biaya transportasi pada akhirnya akan mengurangi biaya produksi keseluruhan, sehingga banyak cara dilakukan untuk menanggulangi masalah transportasi ini [1][2].

Dengan didukung oleh kemajuan teknologi komputasi, telah banyak peneliti yang melakukan penelitian tentang masalah transportasi. Dua masalah transportasi mendasar adalah *Vehicle Routing Problem* (VRP) dan *Vehicle Scheduling Problem* (VSP). Tugas dari kedua masalah ini secara umum adalah untuk membuat pengaturan serangkaian lokasi yang akan dikunjungi oleh sejumlah kendaraan, sehingga biaya tempuh menjadi minimal.

Igarashi mengajukan penggunaan metode *Simulated Annealing* (SA) untuk memecahkan masalah dunia nyata *Vehicle Routing and Scheduling Problem with Single Depot* (VRSP/SD) [1]. Metode ini memberikan hasil yang baik, tetapi terdapat kesulitan dalam pemilihan suhu annealing untuk mencapai solusi optimal. Selain itu, perbedaan unit komputasi yang digunakan dalam evaluasi fungsi energi (berat dan waktu) juga membingungkan pemilihan *weight coefficient* yang tepat.

Chen mengajukan konsep dan formulasi VRSDP/SD (*Vehicle Routing, Scheduling, and Dispatching Problem with Single Depot*) untuk membentangi celah antara metode pencarian konvensional dan situasi kompleks dalam dunia nyata. Model komputasi yang diajukan dalam mengatasi VRSDP/SD memiliki *Hierarchical Multiplex Structure* (HIMS), yang berdasarkan pada

paradigma *object-oriented* dengan metode *tabu search* [2].

Tujuan dari penelitian ini adalah untuk mengajukan metode baru dengan berdasarkan Algoritma Genetika untuk memecahkan masalah VRSP/SD dalam dunia nyata. Perbandingan metode yang diajukan dengan metode terdahulu dilakukan atas data yang sama dari masalah dunia nyata.

Dalam seksi 2 akan dijelaskan tentang konsep VRSP/SD. Seksi 3 memberikan penjelasan singkat mengenai studi terdahulu, sedangkan metode yang diajukan akan dideskripsikan pada seksi 4. Beberapa hasil eksperimen akan ditampilkan pada seksi 5.

2. MASALAH VRSP/SD

2.1. Definisi Masalah VRSP/SD

Masalah pengiriman barang sehari-hari dalam dunia nyata dapat direpresentasikan sebagai masalah VRSP/SD [2] yang dideskripsikan seperti di bawah ini. Suatu depot D (pusat pengiriman, dimana terdapat seluruh barang simpanan) mempunyai L kendaraan $\{V_l\}$ untuk pengiriman barang, dimana terdapat beberapa jenis kendaraan dengan kapasitas yang berbeda. Ketika M pesanan $\{O_m\}$ (kebutuhan akan beberapa barang dalam kurun waktu tertentu) diperoleh dari N pelanggan $\{U_n\}$ (pemesan barang/*user*/*customer*, tiap pelanggan memiliki tempat parkir dan jam kerja sendiri), sebuah jadwal pengiriman untuk memenuhi keseluruhan M pesanan dengan menggunakan R ($\leq$L) kendaraan harus dibuat hingga keesokan harinya. Suatu trip (X_q) dari suatu kendaraan didefinisikan sebagai rangkaian tugas yang harus dikerjakan oleh kendaraan yang bersangkutan, termasuk memuat barang dari depot dan mengantarkan barang-barang tersebut kepada beberapa pelanggan. Masalahnya adalah bagaimana suatu rencana dapat dibuat dengan rute yang optimal dan jadwal (*schedule*) yang efisien untuk keseluruhan tugas pengiriman dan memenuhi beberapa batasan (*constraints*).

Tabel 1 Set Konstanta Dalam Masalah VRSP/SD

Item	Symbol	Number	Universal Set
Depot	D	1	D
User	U_n	N	U={U_1,....U_n,...,U_N}
Order	O_m	M	O={O_1,...,O_m,...,O_M}
Vehicle	V_l	L	V={V_1,.....V_l,.....V_L}

Informasi konstanta yang digunakan dalam masalah VRSP/SD didefinisikan dalam Tabel 1. Deskripsi terperinci adalah sebagai berikut:

$$D = ([Bt_D, Et_D]), \quad (1)$$

dengan integer interval $[Bt_D, Et_D]$ adalah kurun waktu pemuatan barang pada depot.

$$U_n = ([Bt_n, Et_n], SU_n), \quad (2)$$

integer interval $[Bt_n, Et_n]$ adalah kurun waktu jam kerja pelanggan, dan SU_n adalah kapasitas maksimum kendaraan yang dapat masuk ke pelanggan U_n.

$$O_m = (C_m, [Bt_m, Et_m], U_n^m), \quad C_m \leq SU_n^m, \quad (3)$$

C_m adalah kapasitas pemesanan O_m, integer interval $[Bt_m, Et_m]$ adalah kurun waktu pengiriman yang diinginkan, U_n^m adalah pelanggan yang memiliki pesanan O_m, dan SU_n^m adalah ukuran maksimum kendaraan yang dapat memasuki tempat parkir pelanggan U_n^m.

$$V_l = (SV_l, Wt_l, Z_l), \quad (4)$$

SV_l adalah kapasitas maksimum yang dapat dimuat ke dalam kendaraan V_l, Wt_l adalah

awal waktu kerja kendaraan V_l, dan Z_l adalah frekuensi maksimum kendaraan V_l untuk memuat barang (*loading*) dari depot dalam satu hari (berapa kali kendaraan dapat kembali ke depot untuk memuat barang). SU_n, C_m, dan SV_l memiliki nilai integer positif untuk unit yang sama, misalnya berat benda padat (ton). C_m dapat merupakan parameter dengan nilai integer, sementara SU_n dan SV_l adalah nilai integer yang sudah ditetapkan sesuai dengan jenis kendaraan.

Trip X_q dalam masalah VRSP/SD didefinisikan sebagai berikut,

$$J_{mi} = (O_m, U_n^m), \tag{5}$$

$$X_q = ([D, J_{m1}^q, J_{m2}^q, \ldots, J_{mK_q}^q, D], k_q), \tag{6}$$

$$K_q \geq 1, \quad 1 \leq k_q \leq Q,$$

J_{mi} adalah suatu sub-job (dimana jumlah keseluruhan dalam X_q adalah K_q), yang melambangkan informasi pesanan dan batasan pemesan yang bersangkutan, dan k_q adalah nomor index X_q dalam jadwal (Q adalah jumlah keseluruhan trip dalam suatu jadwal).

2.2. Batasan Masalah VRSP/SD

Batasan-batasan berikut ini harus dipenuhi dalam penyelesaian masalah VRSP/SD.

1. Kondisi kendaraan

Kapasitas total pesanan dalam satu trip tidak boleh melebihi kapasitas kendaraan pengangkut.

$$C_{cnst}^q = \begin{cases} 0 \text{ if } \sum_{n=1}^{K_q} C_{mn}^q \leq SV_l \\ 1 \text{ otherwise} \end{cases} \tag{7}$$

$$C_{cnst} = \sum_{q=1}^{Q} C_{cnst}^q = 0 \tag{8}$$

C_{mn}^q adalah jumlah pesanan pelanggan U_n^q dalam trip q, SV_l^q adalah ukuran kendaraan V_l yang bertugas pada trip q, dan Q adalah jumlah keseluruhan trip dalam jadwal.

2. Kondisi Waktu Pengantaran

Setiap pesanan harus diantarkan kepada pemesannya selama jam kerja pelanggan.

$$Ot_{cnst}^q = \begin{cases} 0 \text{ if} \\ Bt_n^q \leq Ot_n^q \leq Et_n^q, \text{ for } \forall n = 1,\ldots,K_q \\ 1 \text{ otherwise} \end{cases} \tag{9}$$

$$O_{cnst} = \sum_{q=1}^{Q} Ot_{cnst}^q = 0 \tag{10}$$

Ot_n^q adalah waktu sampainya pesanan pada *user* U_n, Bt_n^q dan Et_n^q secara berturut-turut adalah waktu permulaan dan akhir jam operasional pelanggan U_n^q.

3. Kondisi Pelanggan

Ukuran kendaraan tidak dapat melebihi besar maksimum tempat parkir yang disediakan pelanggan.

$$U_{cnst}^q = \begin{cases} 0 \text{ if} \\ SV_l \leq SU_n^q, \text{ for } \forall n = 1,\ldots,K_q \\ 1 \text{ otherwise} \end{cases} \tag{11}$$

$$U_{cnst} = \sum_{q=1}^{Q} U_{cnst}^q = 0 \tag{12}$$

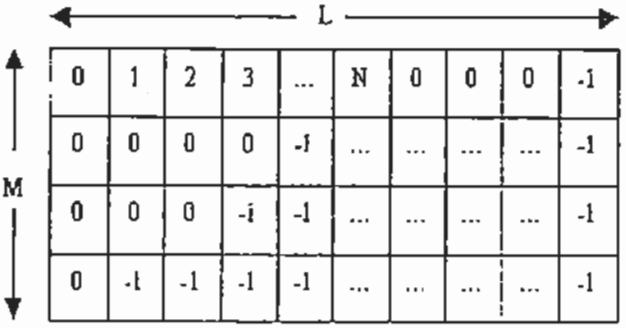
SU_n^q adalah ukuran tempat parkir pelanggan U_n^q dalam trip q.

3. SOLUSI TERDAHULU DALAM PENANGANAN MASALAH VRSP/SD

3.1. Solusi Masalah VRSP/SD Menggunakan *SA*

Seperti telah disebutkan sebelumnya, Igarashi telah menggunakan metode *Simulated*

Annealing untuk memecahkan masalah VRSP/SD.



Gambar 1 Status Awal Matriks 2D

Dalam penelitiannya, setiap jadwal direpresentasikan sebagai matriks dua dimensi (Gambar 1) yang mempunyai ukuran M x L, dimana M adalah jumlah kendaraan yang digunakan, dan L adalah jumlah maksimum lokasi yang dikunjungi dari keseluruhan kendaraan. Setiap elemen matriks merupakan pasangan kendaraan-pelanggan, mempresentasi kan pelanggan mana yang akan dikunjungi oleh sebuah kendaraan. Dengan demikian, suatu baris dalam matriks merupakan urutan pelanggan yang akan dikunjungi oleh satu kendaraan.

Fungsi energi yang digunakan dalam SA didefinisikan sebagai jumlah linear dari batasan dan biaya (*cost*). Setiap batasan dan biaya dikalikan dengan suatu parameter *weight coefficient*. Karena adanya perbedaan unit penghitungan yang digunakan dalam batasan dan biaya tersebut, pemilihan *weight coefficient* yang sesuai untuk keseluruhan faktor merupakan pekerjaan yang membingungkan.

Simulated Annealing dilakukan dengan menukarkan dua elemen matriks berdasarkan jadwal annealing (*annealing schedule*). Penentuan jadwal annealing untuk memecahkan VRSP/SD merupakan pekerjaan yang tidak mudah dan sangat memakan waktu. Oleh karena itu, meskipun penelitian ini membuahkan hasil yang cukup bagus, perlu dilakukan peningkatan untuk mengaplikasikan masalah ini dalam kehidupan sehari-hari.

3.2 Model Perhitungan *Hierarchical Multiplex Structure* (HIMS)

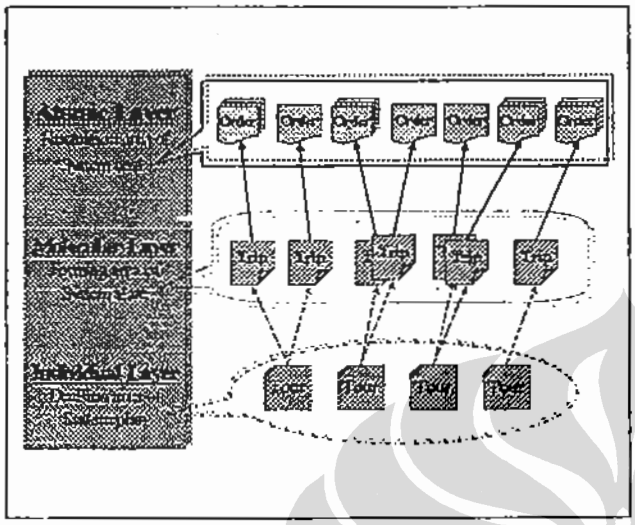
Model HIMS terdiri dari 3 lapisan (Gambar 2), yaitu: lapisan *atomic* (biaya system dapat dikontrol dengan metode heuristik), lapisan *molecular* (status system dapat disesuaikan dengan menggunakan metode heuristik dan penghitungan optimal), dan lapisan *individual* (keseimbangan system dapat dimodifikasi dengan *fuzzy inference*). Model HIMS disusun dengan menggunakan paradigma *object-oriented*, dan *tabu search* digunakan sebagai metode heuristik dalam lapisan *atomic* dan *molecular*. Aplikasi model HIMS dalam penanganan masalah VRSP/SD di dunia nyata ditunjukkan melalui percobaan dengan data yang diambil dari perusahaan minyak [2].

Dalam model HIMS, penelitian ditekankan pada tiga masalah yaitu: *Vehicle Routing*, *Vehicle Scheduling*, dan *Vehicle Dispatching Problem*. Karena masalah *dispatching* juga dipertimbangkan, pada saat kita hendak mengaplikasikan model HIMS untuk menangani masalah VRSP/SD menjadi terlalu rumit. Model HIMS dengan menggunakan lori sebagai alat transportasi memberikan hasil yang baik pada masalah VRSDP/SD, sedangkan implementasi masalah VRSP/SD dengan menggunakan truk sebagai alat transportasi merupakan pekerjaan yang cukup sulit. Dalam dunia nyata transportasi lori, perusahaan pengiriman barang biasanya menggunakan kendaraan dalam jumlah banyak, sehingga masalah *dispatching* sangat perlu dipertimbangkan. Sebaliknya, dalam dunia transportasi truk, perusahaan pengiriman barang biasanya hanya menggunakan relatif sedikit kendaraan, sehingga masalah *dispatching* dapat dihilangkan.

4. SOLUSI ALGORITMA GENETIKA DENGAN MATRIKS 2D

Untuk menyelesaikan masalah VRSP/SD, kami menggunakan metode dengan menggunakan

matriks dua dimensi sebagai representasi jadwal.



Gambar 2 Model HIMS

4.1. Strategi Operasi Algoritma Genetika

Ide dasar operasi Algoritma Genetika dalam penelitian ini adalah dengan mengoperasikan sejumlah (*a group of*) pelanggan, tidak hanya satu pelanggan. Untuk keperluan ini, ada dua jenis *crossover operator* yang digunakan dalam operasi Algoritma Genetika untuk menyelesaikan masalah VRSP/SD, yakni: *move-crossover operator* dan *exchange-crossover operator*. Operator-operator tersebut dijalankan pada trip sebuah jadwal.

4.1.1. Move-Cross Operator

Konstruksi *move-crossover operator* (Gambar 3) didefinisikan sebagai berikut:

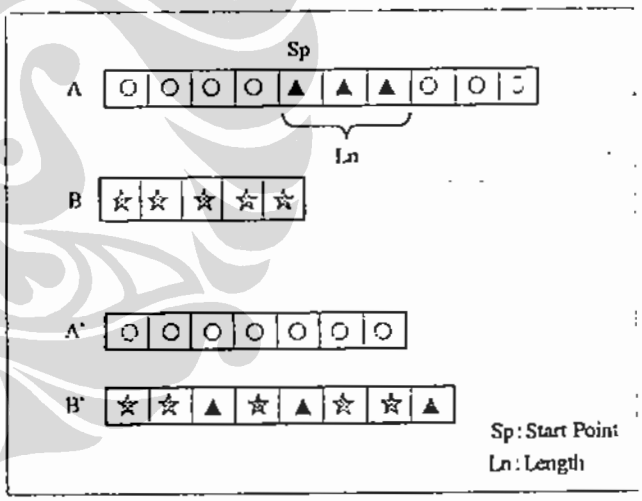
1. Pilih secara acak dari trip A *starting point* (Sp) dan panjang (Ln) dari sub trip, s.t., $1 \le Sp \le N_A$ dan $1 \le Ln \le N_A$ dimana N_A adalah jumlah pelanggan dalam trip A.
2. Pindahkan sub trip A dengan *starting point* Sp dan panjang Ln ke bagian belakang trip B.

Susun kembali trip B dengan menukarkan tiap elemen dan mengaplikasikan metode SA sampai didapatkan rute terpendek. $N_A' = N_A - Ln$ dan $N_B' = N_B + Ln$.

4.1.2. Exchange-Crossover Operator

Konstruksi *exchange-crossover operator* (Gambar 4) didefinisikan sebagai berikut:

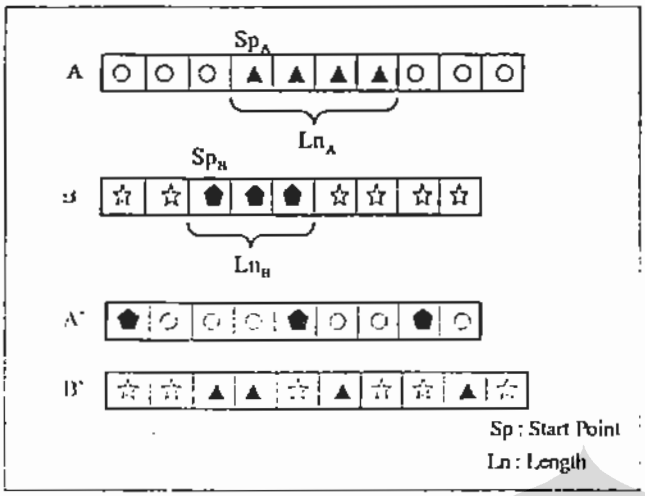
1. Pilih secara acak dari trip A dan B, *starting point* dari masing-masing sub trip (Sp_A dan Sp_B) dan panjangnya (Ln_A dan Ln_B), secara berturut-turut s.t., $1 \le Sp_A \le N_A$, $1 \le Sp_B \le N_B$, $1 \le Ln_A \le N_A$, dan $1 \le Ln_B \le N_B$.
2. Pindahkan sub trip A ke bagian belakang trip B, dan sub trip B ke bagian belakang trip A.
3. Susun kembali trip A dan trip B dengan menukarkan tiap elemen dan mengaplikasikan metode SA sehingga tercapai rute terpendek. $N_A' = N_A + Ln_B$ dan $N_B' = N_B + Ln_A$.



Gambar 3 Operasi *Move-Crossover*

4.2. Fungsi Evaluasi

Sebagai analisis atas kualitas jadwal, kami menerapkan fungsi evaluasi terhadap jadwal yang sudah dibuat. Fungsi evaluasi terdiri dari dua faktor yaitu: *Total Working Time evaluation* dan *Average Loading Capacity Evaluation*.



Gambar 4 Operasi *Exchange-Crossover*

Total Working Time mengevaluasi keseluruhan biaya (*cost*) dari suatu jadwal, yang mencakup biaya perjalanan antara depot dan pelanggan yang dikunjungi, termasuk juga biaya untuk melakukan pengangkutan dan penurunan barang. *Total Working Time evaluation* ditunjukkan melalui persamaan (13).

$$T_{cost} = \sum_{q=1}^{Q} T_{cost}^{q} \qquad (13)$$

$$T_{cost}^{q} = t_{D,u_1}^{q} + \sum_{i=1}^{K_q - 1} t_{u_i,u_{i+1}}^{q} + t_{u_{K_q},D}^{q} + t_{load}^{q} + \sum_{i=1}^{K_q} t_{unloadu_i}^{q} \qquad (14)$$

Persamaan (14) mendefinisikan evaluasi waktu kerja dari trip q, dimana $t_{a,b}^{q}(a,b \in \{U_n\} \cup \{D\}$ adalah biaya perjalanan dari a ke b, K_q adalah jumlah keseluruhan pelanggan yang dikunjungi dalam trip q, t_{load}^{q} adalah biaya untuk memuat barang di depot untuk trip q, dan t_{unload,U_i}^{q} adalah biaya penurunan barang pada pelanggan U_i.

Biaya *loading capacity rate* untuk setiap trip melakukan evaluasi perbandingan antara total jumlah pesanan dengan kapasitas kendaraan pengangkut, seperti ditunjukkan pada persamaan (16). Biaya untuk satu jadwal didefinisikan sebagai rata-rata *loading rate* dari keseluruhan trip (persamaan 15). Nilai C_{cost} yang tinggi lebih dipilih dengan alasan efisiensi.

$$C_{cost} = \left(\sum_{q=1}^{Q} C_{cost}^{q} \right) \Big/ Q \qquad (15)$$

$$C_{cost}^{q} = \left(\sum_{j=1}^{K_q} C_j^{q} \right) \Big/ SV_i^{q} \quad \in [0,1] \qquad (16)$$

4.3. Kriteria Seleksi

Untuk menormalisasikan dua faktor evaluasi yang telah dideskripsikan di atas, kami membuat pengukuran terhadap kedua faktor tersebut. Pengukuran terhadap *working time evaluation* didefinisikan sebagai berikut,

$$g_1(X^{[k]}) = 1 - T_{cost}^{[k]} / T_{cost}^{[0]} \quad \in [0,1] \qquad (17)$$

Sedangkan pengukuran terhadap *Average Loading Capacity Evaluation* didefinisikan sebagai,

$$g_2(X^{[k]}) = C_{cost}^{[k]} \quad \in [0,1] \qquad (18)$$

Dalam persamaan (17) dan (18), *superscript* $^{[0]}$ merupakan nilai awal (*initial value*) dari setiap *objective state*, dan *superscript* $^{[k]}$ merepresentasikan nilai status pada generasi k (hasil sementara). Semakin tinggi nilai g_x *objective state* semakin baik.

Keseluruhan evaluasi kualitas dari semua *objective function* didefinisikan sebagai:

$$g(X^{[k]}) = \rho_1 g_1(X^{[k]}) + \rho_2 g_2(X^{[k]}); \rho_1, \rho_2 \in R^{-} \qquad (19)$$

Untuk membuat keputusan dalam menyimpan status yang lebih baik, suatu kriteria seleksi diterapkan sebagai berikut:

$$eval(k) = \frac{g(X^{[k]}) - g(X^{[0]})}{g(X^{*}) - g(X^{[0]})} \qquad (20)$$

$$g(X^{*}) = \begin{cases} g(X^{[k]}) \; if \; eval(k) > 1 \\ g(X^{*}) \; otherwise \end{cases} \qquad (21)$$

dalam persamaan (20) dan (21), $g(X^*)$ merupakan indikasi nilai *fitness* dari status *local optimal*, sementara $g(X^{[k]})$ merupakan indikasi nilai *fitness* generasi ke k.. Ketika hasil dari eval(k) lebih besar dari 1, status generasi ke k disimpan sebagai nilai optimal.

4.4. Prosedur Algoritma Genetika

Prosedur Algoritma Genetika untuk penyelesaian masalah VRSP/SD ditunjukkan melalui Gambar 5. Dalam sistem ini, setiap jadwal direpresentasikan sebagai matriks dua dimensi.

Pada saat inisialisasi, dua buah baris dipilih dari matriks sebagai *parents*. Pemilihan ini dilakukan secara acak, dimana kedua baris harus merupakan baris yang berbeda.

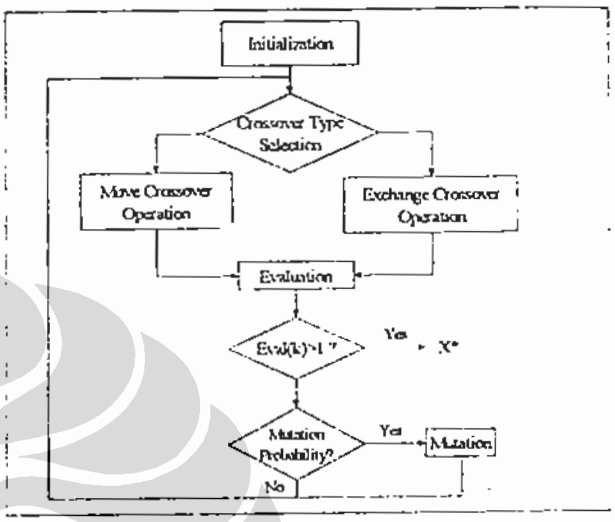
Crossover dilakukan melalui aplikasi *Move* atau *Exchange-Crossover Operation* pada kedua *parent* trip, seperti didefinisikan pada sub seksi 4.2. Pemilihan jenis *crossover* dilakukan secara acak.

Pemilihan jadwal seperti dideskripsikan pada 4.3. dilakukan untuk mendapatkan solusi optimal. Status yang paling baik pada generasi saat itu disimpan pada X*. Status generasi berikutnya dibandingkan dengan X*, menggunakan kriteria seleksi yang telah dijelaskan pada persamaan (20) dan (21).

Operasi mutasi diterapkan melalui penukaran elemen matriks secara acak. Probabilitas mutasi yang digunakan pada makalah ini adalah 0.05.

5. HASIL PERCOBAAN

Percobaan dilakukan dengan menggunakan data nyata perusahaan pengiriman makanan harian dari satu depot menuju 46 toko di daerah Saitama, Jepang (Gambar 6). Kendaraan yang tersedia untuk pekerjaan ini meliputi tiga buah truk dengan kapasitas 4 ton, dan dua buah truk berkapasitas 2 ton. Kombinasi kendaraan tersebut digunakan dalam percobaan, dan hasil dari percobaan ini dibandingkan dengan solusi terdahulu menggunakan SA [1].



Gambar 5 Prosedur Algoritma Genetika

5.1. Percobaan 1

Percobaan 1 dilakukan dengan menggunakan tiga buah truk berkapasitas 4 ton. Setiap kendaraan memiliki kecepatan rata-rata 10 km/jam. *Weight coefficient* ρ1 adalah 100 dan ρ2 adalah 20. Hasil percobaan 1 ditunjukkan pada Tabel 2.

Tabel 2. Hasil Percobaan 1

Vehicle Number	SA		Algoritma Genetika (proposed)	
	Working Time	Loading Rate	Working Time	Loading Rate
4T-01	4:13	0.81	4:31	0.84
4T-02	4:36	0.88	4:26	0.94
4T-03	3:48	0.99	2:57	0.90
Evaluation Value	58.100		61.496	

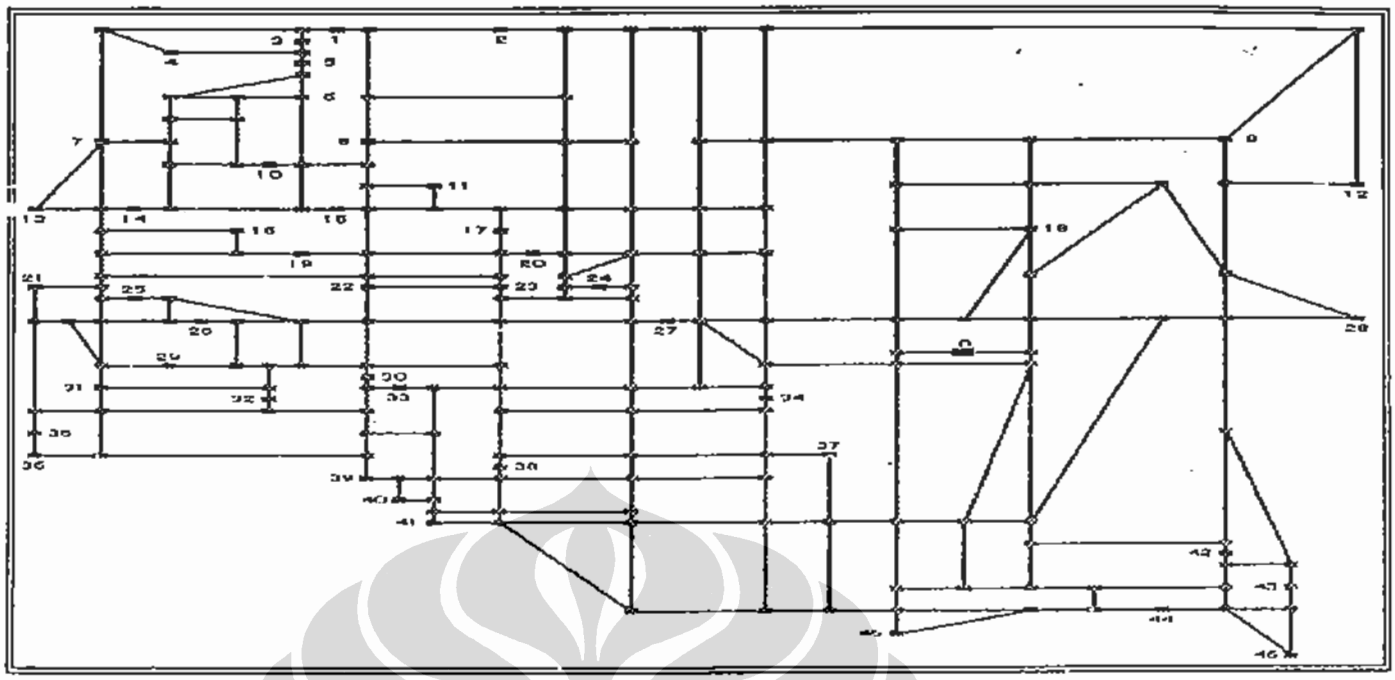


Diagram 6 Peta Digital Pengiriman Makanan Harian

5.2. Percobaan 2

Dalam percobaan 2 tersedia dua buah truk berkapasitas 4 ton dan dua buah truk berkapasitas 2 ton. Kecepatan rata-rata untuk setiap kendaraan adalah sama dengan kecepatan pada percobaan pertama, yaitu 10 km/jam. *Weight coefficient* ρ1 adalah 80 dan ρ2 adalah 30. Hasil percobaan 2 ditunjukkan melalui Tabel 3.

Tabel 3 Hasil Percobaan 2

Vehicle Number	SA		Algoritma Genetika (proposed)	
	Working Time	Loading Rate	Working Time	Loading Rate
4T-01	3:51	0.96	4:27	0.96
4T-02	4:27	0.97	3:41	0.95
2T-01	2:56	0.85	3:01	0.96
2T-02	3:07	0.90	2:22	0.83
Evaluation Value	56.238		59.392	

5.3. Analisis Hasil Percobaan

Dari kedua percobaan di atas, ditunjukkan bahwa terdapat sedikit perbedaan antara metode yang diajukan dan metode terdahulu yang menggunakan SA [1], baik pada factor biaya *working time* maupun *loading rate*. Evaluasi terhadap metode berbasis Algoritma Genetika dapat ditingkatkan jika dibandingkan dengan SA, mengingat besarnya kemungkinan aplikasi Algoritma Genetika untuk penyelesaian masalah VRSP/SD.

6. KESIMPULAN

Masalah VRSDP/SD telah diadaptasi untuk menyelesaikan masalah VRSP/SD dengan menghilangkan masalah *Vehicle Dispatching*. Definisi yang dijelaskan pada makalah ini sesuai untuk masalah VRSP/SD di dunia nyata yang menggunakan truk sebagai alat transportasi.

Matriks dua dimensi digunakan sebagai representasi jadwal, menyebabkan eksplorasi solusi menjadi lebih efisien. Dengan menggunakan matriks 2D, masalah penugasan pelanggan terhadap kendaraan dan masalah mencari rute optimal untuk tiap trip dapat dilihat secara lengkap sebagai masalah VRSP/SD.

Percobaan dilakukan dengan berdasarkan data dari dunia nyata pengiriman makanan harian di daerah Saitama Jepang menunjukkan penerapan metode yang diajukan ini tidak hanya dari sudut pandang secara teori, melainkan juga secara praktis. Dari hasil percobaan, bisa kita lihat bahwa metode yang diajukan dapat menghasilkan jadwal dengan kualitas hampir sama dengan metode sebelumnya yang menggunakan *Simulated Annealing*.

REFERENSI

[1] H. Igarashi, "A Vehicle Schedule Planning System Using a Simulated Annealing Method", *the transactions of the institute of electronics, information and communication engineers*, Japan, vol. J78-D-II, no.5, pp. 819/826 (1995)(in Japanese).

[2]. K. Chen, "Studies of Computational Models with Hierarchical Multiplex Structure for Vehicle Routing, Scheduling, and Dispatching Problems", *Doctoral Thesis*, Department of Computational Intelligence and System Science, Tokyo Institute of Technology (2000).

[3]. L. Bodin, B. Golden, A. Assad, and M. Ball, "Routing and Scheduling of Vehicles and Crews: The State of Art", *Computers and Operations Research*, vol. 10 no. 2, pp. 63/212 (1983)

[4]. I.H. Osman, "Metastrategy simulated annealing and tabu search algorithm for the vehicle routing problem", *Annals of Operations Research*, vol. 41, pp. 421/451 (1993).

[5]. G. Laporte and I.H. Osman: Routing Problems, "A bibliography", *Annals of Operations Research*, vol. 61, pp. 227/262 (1995).

[6]. F. Leclerc, "Genetic algorithms for vehicle dispatching", *Int. Trans. Opl. Res.*, vol. 4, no. 5/6, pp. 391/400 (1997).

[7]. L. Davis, "Genetic algorithms and simulated annealing: an overview", In *L. Davis, genetic algorithms and simulated annealing*, pp. 1/11, London: pitman (1987).

[8]. D. Goldberg, *"Genetic Algorithm in Search, Optimization, and Machine Learning"*, Canada: Addison Wesley (1989).

[9]. M. Gen, R.Cheng; *"Genetic Algorithms and Engineering Design"*, Canada: John Wiley & Sons, Insc. (1997).