

OPTIMASI GENETIC ALGORITHM DENGAN SIMULATED ANNEALING UNTUK MULTIPLE DEPOT CAPACITATED VEHICLE ROUTING PROBLEM

Aditya Permana¹, Mahmud Dwi Sulistiyo², Gia Septiana Wulandari³

^{1,2,3}Prodi S1 Teknik Informatika, Fakultas Informatika, Universitas Telkom

¹adityapermana22@gmail.com, ²mahmuddwis@telkomuniversity.ac.id, ³gia.septiana@gmail.com

Abstrak

Multiple Depot Capacitated Vehicle Routing Problem (MDCVRP) adalah pengembangan dari *Vehicle Routing Problem (VRP)*. Tujuan objektifnya adalah mencari rute kendaraan dengan biaya termurah dari depot ke setiap pelanggan yang harus dikunjungi. Pada MDCVRP, permasalahan bertambah dengan bertambahnya jumlah depot dan adanya *constraint* tambahan berupa batas kapasitas kendaraan. Secara matematis, MDCVRP ini merupakan permasalahan kombinatorial yang termasuk ke dalam kategori NP-hard (*non-deterministic polynomial-time hard*). Pada penelitian ini, digunakan *Genetic Algorithm (GA)* yang dioptimasi oleh algoritma *Simulated Annealing (SA)* dalam menyelesaikan permasalahan MDCVRP. Pada dasarnya, GA sudah cukup bagus dalam menyelesaikan berbagai permasalahan kombinatorial. Namun, kekurangannya adalah adanya kemungkinan GA dalam proses pencariannya terjebak dalam kondisi optimal lokal. Untuk mengatasi kekurangan tersebut, SA hadir untuk mengoptimalkan performansi GA agar terhindar dari konvergensi prematur karena terjebak dalam optimum lokal sehingga hasil yang diperoleh menjadi lebih baik. Observasi telah dilakukan beberapa kali sehingga mendapatkan setting terbaik untuk beberapa parameter yang berpengaruh terhadap sistem. Dari hasil percobaan, terbukti bahwa penerapan SA untuk mengoptimasi GA selalu dapat menaikkan performansi sistem sebesar sekitar 1-2%. Dari keseluruhan pengujian yang telah dilakukan, performansi terbaik yang dihasilkan oleh sistem mencapai 94.51%.

Kata kunci : *Multiple Depot, Vehicle Routing Problem, Genetic Algorithm, Simulated Annealing*

1. Pendahuluan

Lingkungan bisnis global dengan mobilitas yang tinggi menuntut perusahaan pengiriman barang untuk semakin lebih efisien dalam mengatur arus pengiriman barang. Perusahaan pengiriman barang memiliki peran yang sangat penting dalam memastikan kelancaran pengiriman barang yang sangat kompleks. Dengan tingginya ongkos transportasi, dibutuhkan sebuah strategi untuk melakukan pengaturan rute pengiriman barang. Permasalahan ini bisa dikategorikan kedalam permasalahan *Vehicle Routing Problem (VRP)*.

VRP adalah permasalahan di mana terdapat beberapa klien yang berada di lokasi berbeda, kemudian terdapat sebuah kendaraan dari depot yang bertugas untuk mendatangi lokasi dari setiap klien tersebut. Tujuan objektif dari permasalahan VRP adalah bagaimana mengatur rute untuk mendatangi lokasi dari setiap klien sehingga jarak tempuh dengan rute yang dibentuk menjadi seminimal mungkin [6].

Terdapat beberapa jenis permasalahan dari VRP ini, salah satunya adalah yang ditangani pada penelitian ini, yaitu *Multiple Depot Capacitated Vehicle Routing Problem (MDCVRP)*. MDCVRP merupakan permasalahan khusus VRP, di mana jumlah depotnya lebih dari satu dan dengan tambahan *constraint* bahwa setiap kendaraan memiliki batas kapasitas berat.

Secara matematis MDCVRP ini dapat dimodelkan menjadi permasalahan kombinatorial jika diselesaikan dengan pendekatan komputasi. Kompleksitas komputasi dari MDCVRP termasuk ke dalam kelas NP-hard (*non-deterministic polynomial-time hard*), sehingga waktu komputasi yang dibutuhkan untuk melakukan pencarian solusi dengan menggunakan algoritma deterministik akan membutuhkan waktu yang sangat lama [2]. Oleh karenanya, diperlukan sebuah algoritma yang bersifat heuristik untuk memecahkan permasalahan kombinasi pada MDCVRP dengan cepat, seperti *Genetic Algorithm (GA)* dan *Simulated Annealing (SA)*.

GA merupakan algoritma pencarian yang diadaptasi dari proses genetika dan evolusi pada makhluk hidup. Sedangkan SA merupakan algoritma pencarian yang diadaptasi dari bidang metalurgi saat proses pembentukan kristal. Kelebihan pada GA adalah kemampuannya yang baik dan cepat dalam mencari solusi optimal dalam ruang solusi yang sangat besar. Namun, GA memiliki kekurangan yaitu ada kemungkinan terjebak pada optimum lokal sehingga hasilnya akan kurang optimal [3]. Adapun SA dalam proses pencariannya memiliki kemampuan yang baik untuk terhindar dari solusi optimum lokal [5].

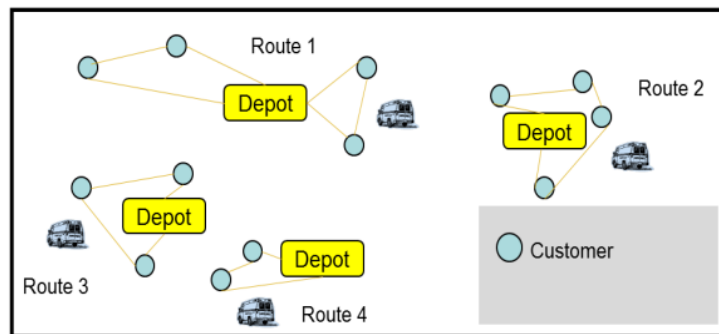
GASA yang merupakan *hybrid* antara algoritma GA dan SA akan digunakan untuk permasalahan MDCVRP ini. Algoritma tersebut menerapkan kelebihan dari SA untuk menutupi kelemahan dari GA. Sebagaimana pada GA, GASA menerapkan operator-operator evolusi, seperti rekombinasi, mutasi dan seleksi individu. Adapun variabel suhu digunakan untuk menentukan jumlah iterasi dan probabilitas pergantian individu terbaik yang sudah mulai jenuh.

Saat penerapan algoritma GASA untuk permasalahan MDCVRP ini, akan muncul permasalahan dalam menentukan nilai parameter-parameter yang berpengaruh pada GA maupun SA. Oleh karena itu, dibutuhkan sebuah observasi untuk mendapatkan nilai parameter-parameter yang optimal untuk GASA. Penentuan nilai dari parameter-parameter tersebut akan mempengaruhi solusi akhir dari GASA dalam menyelesaikan MDCVRP.

2. Multiple Depot Capacitated Vehicle Routing Problem (MDCVRP)

Multiple Depot Capacitated Vehicle Routing Problem (MDCVRP) merupakan variasi dari permasalahan dari VRP yang memiliki lebih dari 1 depot dan diberi tambahan constraint berupa kapasitas angkut kendaraan. Tujuan utamanya adalah sama, yaitu mencari rute terpendek untuk memenuhi semua permintaan pelanggan.

Secara matematis, permasalahan ini dapat digambarkan dengan representasi *undirected graph* $G = (V, E)$, di mana $V = \{0, 1, 2, 3, 4, \dots, N\}$ adalah himpunan *node* yang menunjukkan lokasi dari depot dan pelanggan, sedangkan E merupakan himpunan *edges* yang menunjukkan jalan penghubung antar lokasi.



Gambar 1 Ilustrasi MDCVRP

Tujuannya adalah sebagai berikut, yaitu mencari nilai minimum dari :

$$\text{Min } \sum_{k=1}^K \sum_{i=1}^N \sum_{j=1}^K C_{ij} \cdot V_{ij}^k \quad (1)$$

dan dibatasi oleh *constraint* berikut :

$$\sum_{i=1}^N \sum_{j=1}^N (q_i + q_j) \leq Q^k, 1 \leq k \leq K \quad (2)$$

N adalah jumlah *node*, K adalah jumlah kendaraan yang digunakan, C_{ij} menyatakan jarak antara *node* i dan *node* j . V_{ij}^k menyatakan nilai apakah terdapat rute dari *node* i ke *node* j pada kendaraan ke- k ; jika iya, maka nilainya = 1; sebaliknya nilainya = 0. q_i adalah berat dari barang ke- i dan Q^k adalah kapasitas maksimum dari kendaraan k . Secara matematis, kompleksitas permasalahan dari MDCVRP ini termasuk ke dalam kelas NP-Complete, sehingga jika diselesaikan dengan menggunakan algoritma klasik (berbagai algoritma berbasis deterministik) akan membutuhkan waktu yang sangat lama. Oleh sebab itu, diperlukan solusi menggunakan algoritma heuristik berbasis probabilistik untuk menyelesaikan permasalahan ini dengan lebih efisien.

3. Metode dan Rancangan Sistem

3.1 Optimasi Genetic Algorithm dengan Simulated Annealing

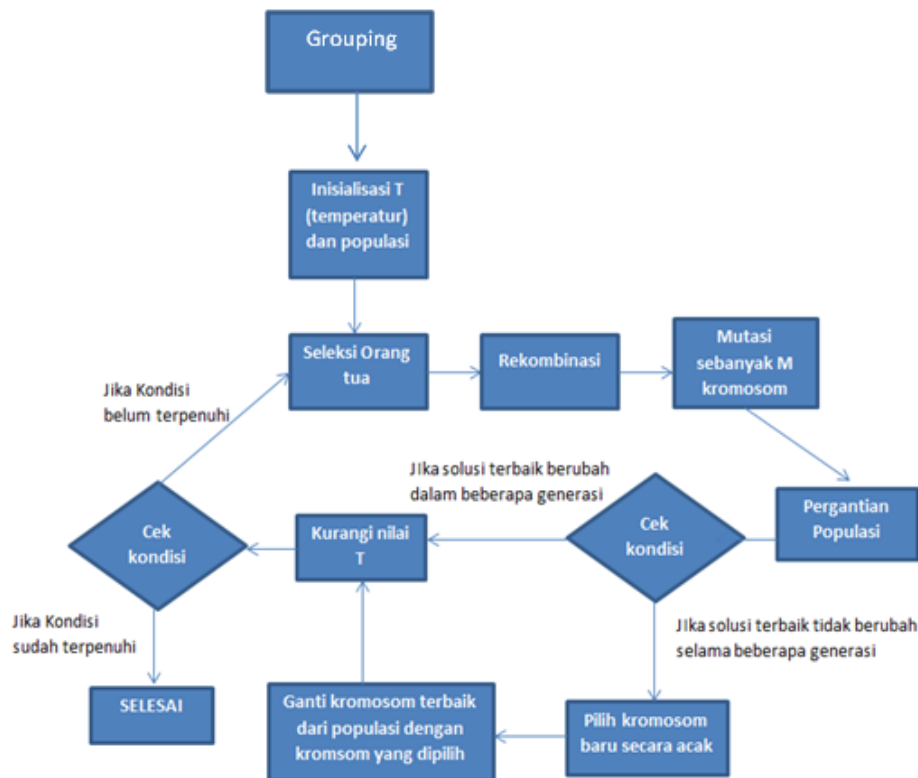
Optimasi *Genetic Algorithm* dengan *Simulated Annealing* (GASA) adalah algoritma yang merupakan *hybrid* antara dua jenis algoritma yaitu *Genetic Algorithm* (GA) dan *Simulated Algorithm* (SA) [8]. Proses dan elemen-elemen yang ada di dalamnya pun berdasarkan kedua algoritma tersebut. GA merupakan algoritma yang pencarian sangat cepat untuk menemukan solusi yang mendekati optimal, meski diberikan ruang solusi yang

sangat besar. Namun, dalam proses pencariannya, ada kemungkinan bahwa GA akan terjebak dalam solusi optimum lokal, sehingga solusi hasil pencariannya tidak akan optimal. Di sisi lain, SA merupakan algoritma pencarian yang memiliki kelebihan untuk bisa keluar dari solusi optimum lokal. Oleh karenanya, SA akan cocok bila dipadukan GA untuk mengoptimalkan hasil pencarian GA.

Berikut adalah algoritma dari GASA yang diterapkan pada penelitian ini [1][8].

1. Inisialisasi T yaitu nilai dari temperatur.
2. Inisialisasi populasi sebanyak N kromosom secara acak.
3. Hitung nilai *fitness* dari masing-masing kromosom.
4. Pilih sebanyak M kromosom dengan metode *Roulette Wheel* dari N kromosom untuk dimasukan ke dalam *mating pool*.
5. Lakukan proses rekombinasi pada kromosom baru hasil rekombinasi yang berada dalam *mating pool* tersebut.
6. Lakukan proses mutasi pada M dengan nilai *fitness* terendah pada kromosom baru dengan dengan probabilitas PM.
7. Hitung nilai *fitness* dari masing-masing kromosom baru.
8. Pilih sebanyak N kromosom terbaik dari N kromosom pada populasi sebelumnya dan dari kromosom-kromosom baru untuk dijadikan sebagai populasi baru.
9. Jika solusi terbaik tidak berubah selama beberapa generasi :
Gantikan kromosom terbaik dari populasi dengan kromosom dengan probabilitas $\exp(-\Delta E/T)$, walaupun kromosom baru ini lebih buruk. ΔE adalah perbedaan nilai *fitness* antara kromosom terbaik dari populasi dan kromosom baru yang dipilih.
10. Kurangi nilai T.
11. Proses selesai jika nilai jumlah iterasi sudah terpenuhi.
12. Ulangi tahap 3.

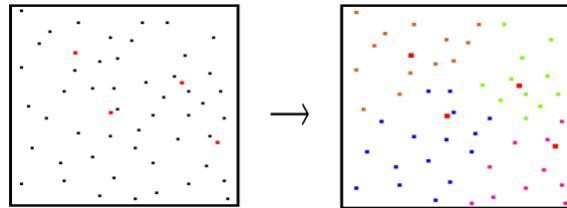
Adapun berikut alur proses yang diterapkan pada sistem yang dibangun.



Gambar 2 Alur Proses pada GASA yang Diterapkan pada Sistem

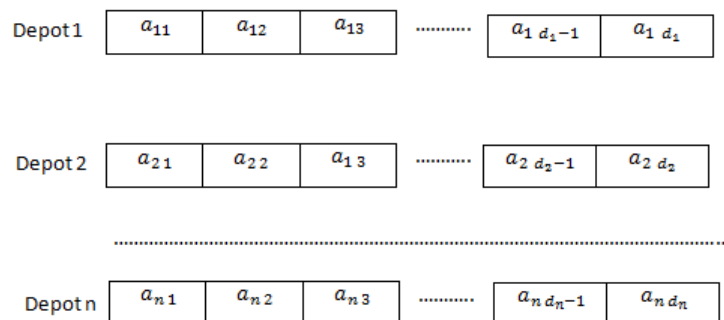
3.2 Grouping

Tahap ini merupakan proses pengelompokan titik-titik customer ke setiap depot yang akan melayaninya. Customer dilayani di suatu depot jika jarak dari customer ke depot tersebut adalah jarak yang terdekat jika dibandingkan dengan jarak ke depot-depot lainnya. Proses ini dilakukan untuk mengefisiensi (mengurangi ruang pencarian) proses pencarian rute terbaiknya.



Gambar 3 Ilustrasi Proses Grouping

Hasil dari proses grouping ini adalah diperolehnya D jenis kromosom (D adalah banyaknya depot) yang akan diproses secara terpisah untuk mendapatkan solusi terbaik.



Gambar 4 Hasil Proses Grouping yaitu D Jenis Kromosom

3.3 Representasi Solusi

Setelah mendapatkan sebanyak D jenis kromosom, setiap jenis kromosom tersebut akan diproses oleh GASA untuk mendapatkan solusi optimalnya. Solusi yang diambil adalah gabungan D kombinasi jenis kromosom yang paling optimal. Panjang kromosom bersifat dinamis karena tergantung jumlah keseluruhan *customer* yang akan dilayani. Representasi kromosom yang digunakan adalah representasi permutasi, yang panjangnya adalah L_d (L_d merupakan jumlah customer yang dilayani oleh depot d).

Sebagai contoh, terdapat sebuah depot d yang berlokasi di (0, 0) dan tabel berikut menunjukkan daftar *customer* mana saja yang akan dilayani oleh depot d tersebut.

Tabel 1 Contoh Input Customer untuk Depot d

Node	q_i	x_i	y_i
3	4	0	7
8	6	0	1
17	1	0	3
22	3	0	9
29	5	0	10
39	2	0	2

Dari tabel 1 di atas, *node* menyatakan id *customer*, q_i menyatakan berat barang yang akan disampaikan ke *node* i, serta x_i dan y_i yang menyatakan lokasi koordinat x dan y dari *node* i. Adapun jarak antar *node* atau $Dist(i,j)$ akan dihitung dengan menggunakan rumus *Euclidean Distance* sebagai berikut.

$$Dist(i,j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \quad (3)$$

Misalnya, masih menggunakan contoh sebelumnya, depot d memiliki 3 truk dengan daya tampung maksimal setiap truk tersebut adalah 8 satuan. Kemudian, misalnya pada iterasi tertentu, terdapat salah satu kromosom yang nilainya sebagai berikut.

39	8	3	22	29	17
a_1	a_2	a_3	a_4	a_5	a_6

Gambar 5 Representasi Kromosom

Hasil dekode dari kromosom di atas, yaitu berupa urutan rute kendaraan yang akan dikunjungi oleh setiap truk dengan cara *greedy*, adalah sebagai berikut. Dari hasil dekode ini, dilakukan perhitungan total *cost* untuk mendapatkan nilai *fitness* dari kromosom yang didekode tersebut.

Tabel 2 Rute Setiap Truk Hasil Dekode Kromosom yang Dicontohkan

Truk	Rute	Total berat	Total Jarak
1	39 - 8	8	4
2	3 - 22	7	18
3	29 - 17	6	20

3.4 Fungsi Fitness

Parameter yang perlu diperhatikan untuk mendapatkan nilai evaluasi setiap individu (*fitness*) pada kasus MDCVRP pada penelitian ini adalah total jarak tempuh yang dilalui oleh kendaraan dan dibandingkan dengan *Best Known Solution* yang informasinya sudah diperoleh dari dataset. Berikut fungsi *fitness* yang dimaksud [4].

$$f = \frac{Bk}{\sum_{k=1}^K \sum_{i=1}^N \sum_{j=1}^K c_{ij} \cdot v_{ij}^k} \cdot 100\% \quad (4)$$

di mana :

f = nilai *fitness*

K = jumlah kendaraan yang digunakan

N = jumlah *node* (depot dan pelanggan)

c_{ij} = jarak antara *node* i dan *node* j

v_{ij}^k = representasi rute kendaraan k , apa ada rute dari *node* i ke *node* j

Bk = *Best Known Solution*

Dari fungsi tersebut dapat kita lihat bahwa semakin kecil total jarak yang ditempuh, maka nilai *fitness* yang didapatkan untuk sebuah kromosom akan semakin besar.

4. Pengujian Sistem

A> Dataset yang digunakan sebagai input pada pengujian sistem ini adalah dataset dari *Cordeau* [7]. Dataset tersebut terdiri dari beberapa atribut, yaitu (1) Jumlah depot dan pelanggan, (2) Jumlah kendaraan, (3) Kapasitas kendaraan, (4) Koordinat lokasi dari tiap depot dan pelanggan, dan (5) Berat barang tiap pelanggan di setiap *node*.

Pada pengujian ini, akan digunakan tiga buah dataset dengan jumlah kendaraan, jumlah *customer*, dan jumlah depot yang berbeda. Ketiga dataset tersebut dipilih karena telah digunakan pada beberapa penelitian sebelumnya yang berkaitan dengan penyelesaian masalah MDCVRP [7].

Tabel 3 Dataset Cordeau yang Digunakan untuk Pengujian Sistem

Jenis dataset	Jumlah Customer	Jumlah Depot	Kapasitas Maksimum Tiap Truk	Jumlah Kendaraan (tiap depot)	Best Known Solution
p01	50	4	80	4	576.87
P03	75	5	140	3	641.19
P06	100	3	100	6	876.50

Tujuan utama dilakukannya pengujian di sini ialah untuk melihat pengaruh diterapkannya SA untuk mengoptimasi performansi yang ditunjukkan oleh GA. Jika hasilnya lebih baik, maka dikatakan SA efektif dalam menyelesaikan salah satu permasalahan pada GA selama proses pencarian solusi. Selain itu, dikarenakan tidak adanya informasi yang baku mengenai setting parameter pada GASA, maka sebelumnya telah dilakukan observasi terhadap nilai-nilai yang terbaik untuk setiap parameter yang berpengaruh pada algoritma GA dan SA. Oleh karenanya, pada pengujian ini nilai-nilai parameter hasil observasi tersebut digunakan.

Tabel 4 Setting Parameter GA

Parameter	Nilai
Jumlah Populasi	100
Seleksi Survivor	Roulette wheel (Linear Ranking)
Crossover	Order Crossover
Mutasi	Scramble Mutation
Λ	20
P_c	1
P_{mut}	0.1
S	1.2

Tabel 5 Solusi Rata-rata GA untuk Masing-masing Dataset

Dataset	Jumlah Generasi	Jarak Tempuh Rata-rata	Fitness Rata-rata	Waktu (detik)	Best Known Solution*
p01	100	620.15	93.02%	0.07	576.87
p03	500	708.6	90.49%	0.31	641.19
p06	2000	1013.5	86.48%	1.12	876.50

Adapun untuk GASA, yaitu GA yang dioptimalisasi dengan SA, berikut nilai-nilai parameter yang digunakan beserta hasilnya.

Tabel 6 Setting Parameter GASA

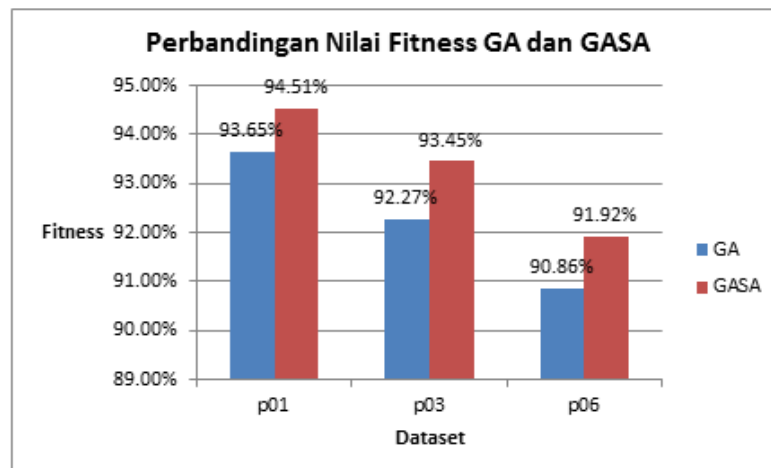
Parameter	Setting
Jumlah Populasi	100
Seleksi Survivor	Roulette wheel (Linear Ranking)
Crossover	Order Crossover
Mutasi	Scramble Mutation
Λ	20
P_c	1
P_{mut}	0.1
S	1.2
T	10000
Cool	0.999
P_g	0.1

Tabel 7 Solusi Rata-rata GASA untuk Masing-masing Dataset

Dataset	Jumlah Generasi	Jarak Tempuh Rata-rata	Fitness Rata-rata	Waktu (detik)	Best Known Solution*
p01	100	622.66	92.65%	0.07	576.87
p03	500	708.16	90.54%	0.4	641.19
p06	2000	1003.29	87.36%	1.33	876.50

Dari tabel-tabel hasil solusi rata-rata yang dihasilkan baik oleh GA maupun GASA, terlihat bahwa *fitness* rata-rata yang diperoleh sudah cukup bagus. Meskipun demikian, jumlah *customer* pada dataset juga ikut berpengaruh, di mana pada p06 yang memiliki jumlah *customer* paling banyak, diperoleh nilai *fitness* rata-rata yang selalu terendah dibandingkan pada kedua data lainnya. Hal ini jelas bahwa semakin banyak jumlah *customer*, maka semakin besar ruang pencariannya, sehingga semakin sulit solusi terbaik untuk ditemukan.

Berikut grafik yang menunjukkan perbandingan hasil terbaik yang diperoleh menggunakan GA maupun GASA untuk masing-masing dataset.



Gambar 6 Grafik Perbandingan Nilai Fitness Terbaik antara GA dan GASA

Dari gambar 6 terlihat bahwa solusi terbaik yang dihasilkan GA yang dioptimalisasi oleh SA selalu lebih baik dibandingkan hanya menggunakan GA untuk semua dataset. Peningkatan performansi berdasarkan hasil percobaan tersebut terlihat sebesar sekitar 1-2% untuk masing-masing dataset.

5. Kesimpulan dan Saran

Kesimpulan dari penelitian ini adalah sebagai berikut.

1. Berdasarkan percobaan yang telah dilakukan, *Genetic Algorithm* terbukti mampu mencari solusi untuk kasus MDCVRP dengan nilai *fitness* diatas 90% untuk semua dataset yang tersedia, di mana hasil terbaik yang diperoleh mencapai 93.65%.
2. Berdasarkan percobaan yang telah dilakukan, algoritma *Simulated Annealing* terbukti mampu meningkatkan performansi *Genetic Algorithm*, sehingga hasil yang diperoleh naik sekitar 1-2%. Adapun hasil terbaik yang diperolehnya adalah 94.51%

Untuk pengembangan sistem lebih lanjutnya dapat dilakukan hal berikut.

1. Dengan optimasi *Simulated Annealing* pada *Genetic Algorithm*, maka perlu dicoba untuk mengoptimalkan *Genetic Algorithm* dengan algoritma optimasi lainnya sehingga hasilnya dapat lebih baik lagi.
2. Pada kasus nyata, *fairness* dari rute yang ditempuh oleh setiap truk adalah hal yang perlu diperhatikan, karena hal ini mampu mengoptimalkan waktu selesainya pengiriman barang ke customer.
3. Meminimalisasi jumlah penggunaan truk yang digunakan akan mampu memangkas biaya dari pengiriman semua barang ke semua customer.

Daftar Pustaka:

- [1] Tamilarasi and kumar, T. Anantha. 2010. *An Enhanced Genetic Algorithm with Simulated Annealing for Job-shop Scheduling*. International Journal of Engineering, Science and Technology, Vol. 2, No. 1, 2010, pp. 144-151.
- [2] Bashiri, Mahdi and Fallahzade, Ehsan. 2012. *A Particle Swarm Optimization Algorithm For Multi-Depot Capacitated Location-routing Problem With Inventory Decision In Supply Chain Network Design*. CIE42 Proceedings, 16-18 July 2012, Cape Town, South Africa © 2012 CIE & SAIIE.
- [3] D.E. Goldberg. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley. Reading, Massachusetts.
- [4] Reno Aditya Putra. 2011. *Penyelesaian Capacitated Vehicle Routing Problem Menggunakan Algoritma Evolution Startegies*. Fakultas Informatika IT Telkom, Bandung, Indonesia.
- [5] S. Kirkpatrick, C.D. Gelatt Jr., M.P.Vecchi. *Optimization by Simulated Annealing*. Science 220 (4598) (1983) 671-680.
- [6] Sombuntham, Pandhapon and Kachitvichayanukul, Voratas.2010. *A Particle Swarm Optimization Algorithm for Multi-depot Vehicle Routing problem with Pickup and Delivery Requests*. International MultiConference of Engineers and Computer Scientists 2010 Vol III, IMECS 2010, March 17 - 19, 2010, Hongkong.



- [7] Surekha P and S.Sumathi. *Solution To Multi-Depot Vehicle Routing Problem Using Genetic Algorithms. World Applied Programming*. Vol (1), No (3), August 2011. 118-131, 2010.
- [8] Suyanto. 2008. *Evolutionary Computation: Komputasi Berbasis Evolusi dan Genetika*. Penerbit Informatika, Bandung.