Penyelesaian Vehicle Routing Problem with Time Windows (VRPTW) dengan Modified Differential Evolution Algorithm

Heri Awalul Ilhamsah[1]
[1]Jurusan Teknik IndustriUniversitas Trunojoyo Madura
E-mail: hilhamsah@yahoo.com

ABSTRAK

Penelitian ini membahas modifikasi algoritma Differential Evolution untuk menyelesaikan permasalahan Vehicle Routing Problem with Time Windows (VRPTW). Pengembangan algoritma dilakukan dengan jalan menambahkan teknik pembangkitan inisial solusi. Teknik pembangkitan insial solusi yang pertama adalah dengan menggunakan fungsi random,kemudian menggunakan neighbor berdasarkan nearest distance (jarak terminimum). Sedangkan teknik pembangkitan solusi selanjutnya adalah dengan insersi solomon. Hasil penelitian ini mengkonfirmasikan bahwa pengembangan algoritma yang dilakukan mampu menemukan solusi yang sama dengan best known solusi dari data yang digunakan sebagai data uji, baik dari jumlah kendaraan yang digunakan ataupun jarak yang dihasilkan. Algoritma modified differential evolution mampu bekerja kompetitif pada data test solomon C105, C106, C107, C108 dan C109 dengan nilai gap sebesar 0%.

***Kata kunci**: algoritma modified differential evolution, vrptw, random, nearest neighbor, insersi solomon.*

ABSTRACT

This studydiscusses modification ofthe DifferentialEvolutionalgorithmto solve theVehicleRoutingProblem withTimeWindows(VRPTW). Algorithmdevelopmentis done byadding theinitialsolutiongeneratingtechnique. First initials solution generationtechniqueis use arandomfunction, then based onnearestneighbordistance (minimum distance). The next initials solution generationtechniqueis use solomon insertion. These resultsconfirmthat thedevelopment of algorithmscapable findingsolutionsthatdothe samewith thebestknownsolutions fromthe data usedasdata test, eitherthe number ofvehicles usedorthe resultingdistance. Modifieddifferentialevolutionalgorithmis able to workcompetitivelyin the solomon data test C105, C106, C107, C108 andC109with gapvalueof 0%.

***Keyword**: modified differentialevolution algorithm, vrptw, random, nearestneighbor, solomon insertion*

PENDAHULUAN

Salah satu bentuk pengembangan persoalan Vehicle Routing Problem (VRP) standar adalah persoalan VRPTW dengan menambahkan batasan waktu. Batasan waktu ini bisa dari operasioanl depot atau operasional konsumen. Sehingga, proses distibusi barang selain memperhatikan kapasitas kendaraan angkut juga tidak boleh melanggar batasan waktu yang telah di tetapkan. Rute yang terbentuk pada persoalan VRPTW berawal dan berakhir pada depot yang sama dengan jalur subrute yang berbeda untuk masing-masing kendaraan.

Persoalan VRPTW masuk dalam kategori NP-hard (Savelsbergh, 1985), sehingga penyelesaian permasalahan tersebut secara eksak sangatlah tidak efisien karena membutuhkan waktu komputasi yang besar (Desrochers, & Solomon, 1992). Untuk itu diperlukan pendekatan lain guna mengirangi besarnya waktu komputasi yang dibutuhkan dalam menyelesaikan persoalan VRPTW. VRPTW masuk dalam ketegori persoalan optimasi, teknik lain selain metode eksak untuk pencarian solusinya bisa menggunakan prosedur pencarian heuristik (heuristics). Heuristik merupakan suatu teknik pencarian solusi dengan sedikit mengabaikan apakah solusi yang di dapat tersebut presisi. Sehingga solusi yang didapatkan bukan merupakan solusi sebenarnya tetapi mendekati solusi sebenarnya. Semakin dekat solusi yang didapatkan dengan nilai sebenarnya maka semakin baik teknik heuristik yang digunakan.

Penggunaan teknik heuristik dimaksudkan untuk mendapatkan hasil yang secara komputasi lebih cepat dengan konsekuensi mengurangi kepresisian atau keakuratan hasil solusi tadi. Jadi kecepatan penghitungan biasanya lebih baik (dibandingkan optimasi eksak) dengan sedikit mengorbankan akurasi solusi yang didapatkan. Pendekatan heuristik ini sifatnya spesifik sehingga untuk persoalan tertentu diperlukan teknik heuristik lainnya. Salah satu bentuk pengembangan dari teknik heuristik adalah metaheuristik, yaitu bentuk pencarian solusi yang memadukan prosedur pencarian dengan strategi tertentu agar dapat keluar dari solusi yang sifatnya lokal optima sehingga dihasilkan solusi yang sifatnya global optima.

Penelitian di bidang VRPTW yang menggunakan teknik metaheuristik untuk prosedur pencarian solusi dilakukan oleh (Berker dan Barkaoui, 2004) *paralel hybrid genetic algorithm*. Paraskevopoulos et, al. (2007) menggunakan *neighborhood tabu search hybrid metaheuristic algorithm* untuk menyelesaikan persoalan VRPTW. Yu et, al (2011) memadukan *ant colonyalgorithm* dengan *tabu search* untuk menyelesaikan persoalan VRPTW. Beberapa metode tersebut mampu mendapatkan hasil rute dengan jarak yang lebih minimum pada data test s*olomon prolem*, dibandingkan dengan solusi yang selama ini diketahui.

Differential evoluion merupakan salah satu metode pencarian solusi dalam keluarga metaheuristik. Mingyong dan Erbao (2010) mengaplikasikan algoritma *differential evolution* (DE) untuk menyelesaikan persoalan *vehicle routing problem delivery pick up with time windows* (VRPDPTW). Dalam penelitian tersebut modifikasi algoritma yang dilakukan. Mingyong dan Erbao (2010) menambahkan kontrol parameter pada penentuan nilai fraksi mutasi (F) dan probabilitas *cross over* (CR). Nilai kedua parameter tersebut akan meningkat selaras dengan jumlah iterasi yang dilakukan. Tasgetiren et, al. (2007) menggunakan *differential evolution algorithm* untuk menyelesaikan permasalahan penjadwalan mesin.

METODA

Modifikasi algoritma DE dilakukan dengan menambahkan proses pembangkitan inisial solusinya. Pada algoritma DE murni pembangkitan solusi dikendalikan oleh fungsi random. Modifikasi yang dilakukan berupaya menambahkan teknik pembangkitan inisial solusi dengan *nearest neighbor* berdasarkan *earliest open* dan pembangkitan inisial solusi berdasarkan insersi Solomon. Algoritma DE akan beruapaya mengurangi jarak dari solusi yang dihasilkan oleh ketiga metode pembangkitan solusi tersebut. Tahap-Tahap dalam algoritma DE yang dikembangkan adalah sebagai berikut :

Tahap Pembangkitan Inisial Solusi dengan Fungsi Random

Tahap pembangkitan solusi dengan fungsi random dijalankan dengan membangkitkan bilangan random antara 1-0. Bilangan random dibangkitkan dengan fungsi rand, dimana bilangan yang dihasilkan terletak antara (0, 1). Angka 0 menunjukkan batas bawah dari bilangan random yang dihasilkan sedangkan angka 1 menunjukkan batas atasnya. Indeks j menunjukkan variabel ke j. Dalam kasus minimasi fungsi dengan 2 variabel, maka j akan bernilai 1 dan 2. Penentuan batas atas dan batas bawah sangat tergantung pada permasalahan yang dihadapi. Jika nilai yang dicari sulit ditentukan posisinya, maka rentang batas atas dan batas bawah bisa dibuat lebih lebar, atau sebaliknya jika titik-titik calon solusi sudah bisa di duga maka batas atas dan batas bawah bisa dipersempit.

$$x_{j,i,0} = lbj + randj\ (,\ 1)(ubj - lbj\ )$$

Tahap Pembangkitan Inisial Solusi dengan *Nearest Neighbor*

Pembangkitan solusi dengan teknik *nearest neighbor* di dasarkan atas jam buka konsumen yang lebih awal. Jadi inisial solusi didapatkan dengan menginsersikan *node* yang memiliki jam buka lebih awal

diantara depot dan *nodecustomer* dengan jam buka paling akhir. Sekumpulan jalur *node* tersebut selanjutnya akan dievaluasi agar tidak melanggar konstrain kapasitas.

Tahap Pembangkitan Inisial Solusi dengan *Insersi Solomon*

Pembangkitan solusi dengan insersi Solomon mengacu pada algoritma insersi yang diusulkan oleh Solomon (1987) yang dapat diilustrasikan sebagai berikut :

Notasi :

t_j' = waktu mulai pelayanan di node j setelah dilakukan insersi

t_j = waktu mulai pelayanan di node j sebelum dilakukan insersi

d_{iu} = jarak antara node i dan u (*unroutedcustomer*)

d_{uj} = jarak dari u (*unroutedcustomer*) ke node j

d_{ij} = jarak dari node i ke node j

Langkah – langkahnya adalah sebagai berikut :

1. Buatlah *seedroute* yang berisi satu *customer*

   Contoh : {1 4 1} dalam hal ini *customer* no 4 adalah *seedcustomer*, dengan 1 sebagai depot.*Seedcustomer* dapat dipilih dengan pertimbangan :
   - Memiliki jarak terjauh dari depot, atau
   - *Customer* dengan *earliest due date* (paling awal jam tutupnya), atau
   - Selang – seling jarak terjauh maupun *earliest due date*
2. Identifikasilah letak insersi untuk seluruh *customer* yang belum masuk ke rute, dengan syarat bahwa letak insersi tersebut adalah *feasible*

   taruhlah :

$$\{v_0 = 0, v_1, v_2, \dots, v_m, v_{m+1} = 0\}$$

   sebagai rute sekarang, dengan

   v_0 = depot

   v_{m+1} = depot
3. Hitung posisi insersi terbaik, yakni

$$c_1(u) = \min_{q=0,\dots,m} c_1(v_q, u, v_{q+1})$$

   dengan syarat bahwa insersi u diantara 2 node yakni v_q dan v_{q+1} adalah *feasible*, *u* adalah *unroutedcustomer*.

   Kriteria c_1 dihitung dengan cara :

$$c_1(i,u,j) = \alpha_1 . \delta_d + \alpha_2 . (t_j' - t_j)$$

   dimana

$$\delta_d = d_{iu} + d_{uj} - \mu . d_{ij}$$

   dengan

$$\alpha_1 + \alpha_2 = 1$$
$$\alpha_1, \alpha_2, \mu \geq 0$$
4. Apabila nilai c_1* sudah dihitung untuk setiap *customer*, maka hitung nilai c_2 untuk setiap *customer* yang memiliki nilai c_1*. (*Customer* yang tidak bisa di-insersi di ruang manapun, maka tidak memiliki nilai c_1 dan demikian tidak memiliki nilai c_1* dan c_2)

   Kriteria c_2 dihitung dengan cara :

$$c_2(u) = \lambda . d_{0u} - c_1(u)$$

   dengan $\lambda \geq 0$
5. Pilihlah *customer* yang memiliki nilai c_2 paling maksimum untuk di-insert-kan. Insert *customer* tersebut ke dalam rute yang sedang dibentuk, sesuai dengan posisi terbaiknya yang ditunjukkan c_1

$$c_2(i(u^*), u^*, j(u^*)) = \text{optimum}\,[c_2(i(u), u, j(u))],$$

   Dengan catatan bahwa u belum masuk ke rute dan insersi bersifat *feasible*
6. Apabila rute yang sedang dibangun sudah tak lagi bisa dilakukan insersi, maka buatlah rute baru (*seedroute*), dan ulangi algoritma (kembali ke langkah 1)

7. Apabila semua *customer* telah berada dalam rute (tidak ada yang tidak terlayani), maka hentikan proses, dan lanjutkan ke sub tahapan *routeminimization*.

Mutasi

Setelah tahapan inisialisasi, DE akan memutasi dan mengkombinasi populasi awal untuk menghasilkan populasi dengan ukuran N vektor percobaan. Dalam DE, mutasi dilakukan dengan cara menambahkan perbedaan dua vektor terhadap vektor ketiga dengan cara random.

$$v_{i,g} = x_{r0,g} + F.(x_{r1,g} - x_{r2,g})$$

Faktor skala, F Є (0, 1+) adalah bilangan *real* positif yang mengendalikan tingkat pertumbuhan populasi.

Crossover

Pada tahap ini DE menyilangkan setiap vektor, $x_{i,g}$, dengan vektor mutan, $v_{i,g}$, untuk membentuk vektor hasil persilangan, $u_{i,g}$.

$$u_{i,g} = u_{j,i,g} = \begin{cases} v_{j,i,g} & if\ (rand_j(0,1) \le Cr, \quad or \quad j = j_{rand}) \\ x_{j,i,g} & sebaliknya \end{cases}$$

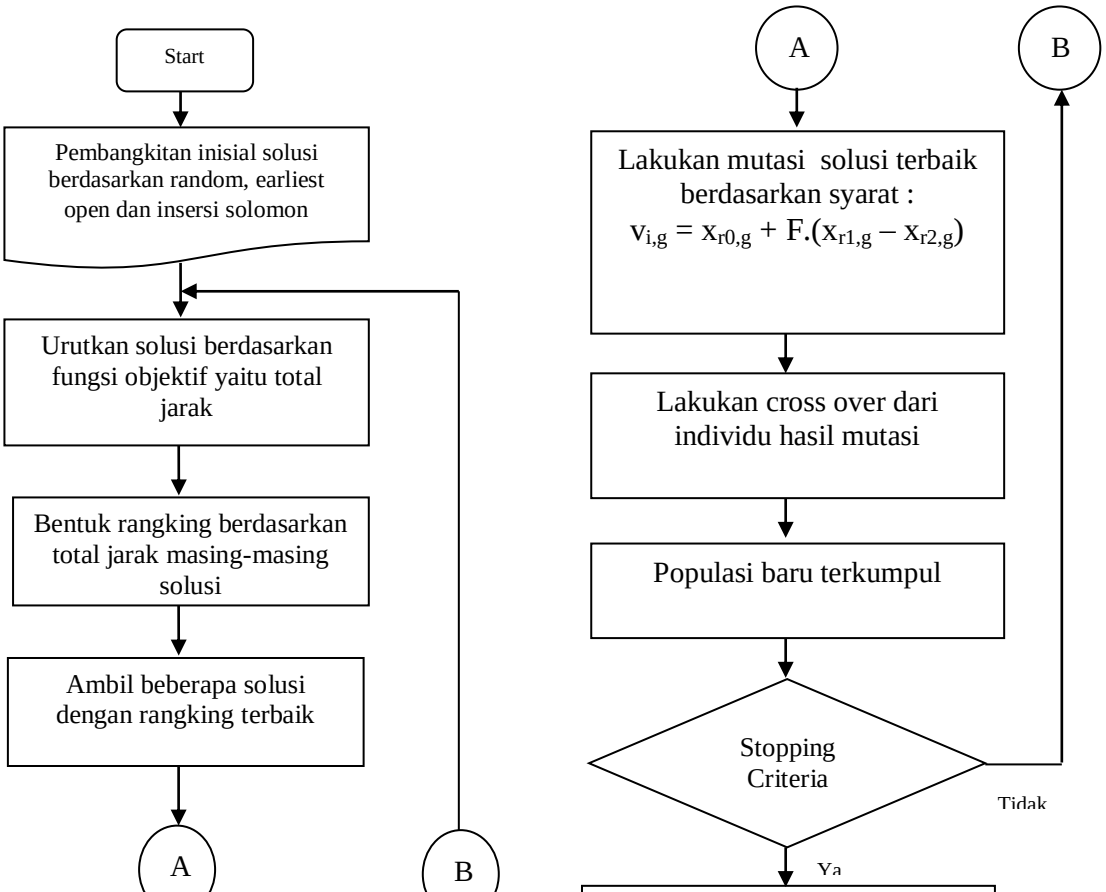
Probabilitas crosover, *Cr Є (0,1)* adalah nilai yang didefinisikan untuk mengendalikan fraksi nilai parameter yang disalin dari mutan.

Selection

Jika trial vector, $u_{i,g}$, mempunya fungsi tujuan lebih kecil dari fungsi tujuan vektor targetnya, $x_{i,g}$, maka ui,g akan menggantikan posisi $x_{i,g}$ dalam populasi pada generasi berikutnya. Sebaliknya target akan tetap pada posisinya dalam populasi.

$$x_{i,g+1} = \begin{cases} u_{i,g} & if\ (f(u_{i,g}) \le f(x_{i,g}) \\ x_{i,g} & sebaliknya \end{cases}$$

Proses akan diulang sampai *stopping criterion* tertentu dicapai. Berikut disajikan alur algoritma *differential evolution.*



Gambar 1. Alur Algoritma *Modified Differential Evolution*

PEMBAHASAN

Modifikasi algoritma yang dilakukan akan diujicobakan pada problem Solomon dengan kode data C101, C102 sampai C109 jumlah *node customer* sebanyak 100 titik.*Setting* parameter untuk algoritma *differential evolution* menggunakan jumlah populasi sebanyak 100, dengan F sebesar 0.8 dan Cr sebanyak 0.3. Hasil penyelesaian problem tersebut akan dibandingkan dengan *best known solution* yang dapat dilihat di http://w.cba.neu.edu/~msolomon/problems.htm. Parameter yang dibandingkan adalah jumlah kendaraan yang dibutuhkan dan jarak yang dihasilkan oleh teknik solusi *modified differential evolution*. Melalui pembandingan tersebut akan dapat diketahui kinerja algoritma dalam menemukan solusi terhadap permasalahan yang ada. Representasinya dapat dilihat dari jarak/gap solusi yang didapatkan dengan *best known solusinya*. Untuk menghitung gap digunakan perhitungan dengan cara dibawah :

$$Gap = \frac{solusi\ algoritma - solusi\ acuan}{solusi\ acuan} x\ 100\%$$

Semakin kecil gap yang dihasilkan dengan *best known solusinya* maka semakin kompetitif algoritma yang kembangkan.

Hasil Komputasi

Hasil *running* algoritma dalam menemukan solusi dapat dilihat pada tabel dibawah :

Tabel 1. Hasil runing algoritma *modified differential evolution*

Tipe	Problem	*Best Known Solusi*		Solusi Oleh *Modified Differential Evolution*		Gap
		Jumlah Kendaraan	Jarak	Jumlah Kendaraan	Jarak	
C1	C101	10	828.94	10	828.96	0.0024%
	C102	10	828.94	10	854.62	3.0979%
	C103	10	828.06	10	857.65	3.5734%
	C104	10	824.78	10	874.76	6.0598%
	C105	10	828.94	10	828.94	0.0000%
	C106	10	828.94	10	828.94	0.0000%
	C107	10	828.94	10	828.94	0.0000%
	C108	10	828.94	10	828.94	0.0000%
	C109	10	828.94	10	828.94	0.0000%

Dari hasil *running* diatas untuk 100 titik *customer*, algoritma *modified differential evolution* mampu bekerja dengan baik pada data C105,106 sampai C109, hal tersebut ditunjukkan tidak adanya gap antara solusi algoritma yang dikembangkan dengan *best known solusinya*. Sedangkan pada data C101,102 dan C103 diperoleh deviasi jarak yang cukup besar. Deviasi ini disebabkan algoritma yang dikembangkan bekerja kurang optimal pada karakteristik data dengan variansi yang sangat tinggi. Koordinat titik konsumen dengan jarak yang besar menyebabkan kinerja algoritma kurang kompetitif. Rata-rata gap yang dihasilkan untuk menyelesaikan permasalahan tersebut sebesar 1.41%. Adanya teknik pembangkitan inisial solusi insersi solomon pada algoritma yang digunakan memberi pengaruh positip dalam upaya peningkatan kinerjan algoritma.

KESIMPULAN

Hasil komputasi menunjukkan algoritma yang dikembangkan mampu menemukan solusi dengan nilai deviasi sebesar nol dengan best known solusi untuk beberapa data test yang digunakan, dengan demikian algoritma yang dikembangkan telah menunjukkan kinerja kompetitif dalam menyelesaikan persoalan VRPTW.

DAFTAR PUSTAKA

Berger, J., Barkaoui, M. (2004). A parallel hybrid genetic algorithm for the vehicle routing problem with time windows. Computers & Operations Research, 31, 2037–2053.

Jin, A.T., Kachitvichyanukul, V. (2008). Particle Swarm Optimization and Two Solution Representations For Solving The Capacitated Vehicle Routing Problem. Elsevier-Science Direct.

Mingyong, L., Erbao, C. (2010). An improved differential evolution algorithm for vehicle routing problem with simultaneous pickups and deliveries and time windows. Journal of Engineering Applications and Artificial Intelligence, 23, 188–195.

Nagy, G. Salhi. S.(2005).Heruistic Algorithm for Single and Multiple Depot Vehicle Routing Problems with Pickup Delivery. European Journal of Operational Research (162), 126-141.

Repoussis, P.P.,Tarantilis C.D., (2010). Solving the fleet size and mix vehicle routing problem with time windows via adaptive memory programming. Journal of Transportation Research Part C. 18, 695-712

Stron, R., Price, K. (1997). Differential Evolution a Simple and Efficient Heuristic for Global Optimization over Continuous Space. Journal of Global Optimization (11), 341-359.

Savelsbergh, M. (1985). Local search in routing problems with time windows. Journal of Operations Research, 4, 285–305.

Solomon, M. (1987) Algorithms for the vehicle routing and scheduling problems with time window constraints. Journal of Operations Research, 35 (2) 254–65.

Tasgetiren, M.F., Pan, Q.K.,Liang, Y.C., Sugantan, P.N., (2007). A discrete differential evolution for the total earliness and tardiness penalties with a common due date on a single-machine. Proceeding of the IEEE Symposium on Computational Intelligence in Scheduling.

Tsai, J.T., Ho, W.H., Liu,T.K., dan Chou, J.H. (2007). Improved Immune Algorithm for Global Numerical Optimization and Job-Shop Scheduling Problem. Jurnal Applied Mathematics and Computation (194), 406-424.

Tan,K.C., Lee, L.H., Zhu, Q.L. dan Ou, K. (2001). Heuristic Methods for Vehicle Routing Problem with Time Windows. Journal Artificial Intelligence in Engineering (15),281-295.

Yu, B.,Yang Z.Z.,Yao B.Z., (2011). A Hybrid algorithm for vehicle routing problem with time windows. Journal of Expert System with Applications, 38, 435-441.