

BAB 2

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1. *Vehicle Routing Problem*

Vehicle Routing Problem merupakan permasalahan distribusi yang mencari serangkaian rute untuk sejumlah kendaraan dengan kapasitas tertentu dari satu atau lebih depot untuk melayani konsumen. Toth dan Vigo (2002) mengemukakan tujuan yang ingin dicapai dalam VRP di antaranya:

- a. Meminimalkan ongkos perjalanan secara keseluruhan yang dipengaruhi oleh keseluruhan jarak yang ditempuh dan jumlah kendaraan yang digunakan.
- b. Meminimalkan jumlah kendaraan yang digunakan untuk melayani semua konsumen.
- c. Menyeimbangkan rute.
- d. Meminimalkan keluhan pelanggan.

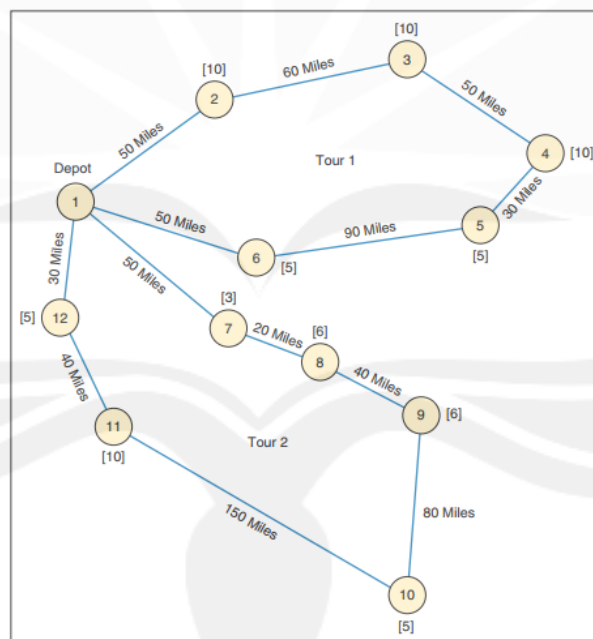
Permasalahan VRP biasanya digambarkan dalam sebuah grafik. Grafik tersebut menggambarkan permasalahan yang terjadi, yaitu berupa penyebaran konsumen yang harus dilayani dan posisi depot yang merupakan pusat pendistribusian berlangsung. *Vertex* ($V_0, \dots, V_n$) merupakan titik yang menunjukkan posisi depot dan konsumen berada. *Vertex* depot ditunjukkan oleh V_0 dan yang lainnya menunjukkan konsumen yang berjumlah n . Garis yang menghubungkan antar *vertex* disebut *arc*. *Arc* menunjukkan waktu, ongkos perjalanan dan jarak yang digunakan untuk perjalanan dari satu titik ke titik yang lain. Solusi dianggap layak jika memenuhi beberapa syarat, yaitu rute yang terbentuk harus dapat melayani semua konsumen, semua konsumen hanya bisa dikunjungi satu kali, dan semua rute harus dimulai dan selesai di *home depot* (Haksever *et al.*, 2000). VRP memiliki karakteristik berupa *demand* yang berada pada setiap konsumen, memiliki satu depot, dan memiliki lebih dari satu kendaraan dengan kapasitas yang terbatas. Gambar 2.1. menunjukkan contoh penyelesaian VRP dengan menghasilkan 2 rute dan masing-masing rute kembali ke titik awal (depot). Rute 1 melayani 5 konsumen dan rute 2 melayani 6 konsumen dengan jumlah permintaannya masing-masing tanpa melanggar *load* maksimum yang mampu dibawa oleh setiap kendaraan.

VRP diklasifikasikan dalam *NP-hard problem*, oleh karena itu metode *exact optimization* sulit untuk menyelesaikan kasus VRP. Untuk mendapatkan solusi yang relevan dengan kondisi *real* dan sangat dekat dengan solusi yang optimal

maka digunakanlah metode *heuristic* dan *meta-heuristic* (Kumar dan Panneerselvam, 2012).

VRP memiliki banyak varian sesuai dengan karakteristik permasalahan, salah satunya adalah *Capacitated Vehicle Routing Problem (CVRP)*. CVRP pertama kali muncul pada permasalahan penentuan rute distribusi dengan kendaraan yang memiliki kapasitas tetap dan sama untuk memenuhi permintaan konsumen dengan komoditi tunggal dari suatu depot dengan ongkos minimum (Ai dan Kachitvichyanukul, 2009). CVRP merupakan varian VRP yang paling sederhana. Kapasitas yang terbatas dan sama untuk semua kendaraan merupakan ciri dari CVRP.

CVRP digambarkan dengan sejumlah *demand* (q_i) yang harus dikirimkan ke konsumen i ($i = 1, 2, \dots, n$) dari depot tunggal menggunakan armada pengiriman dengan kapasitas C . Dalam CVRP setiap konsumen hanya boleh dikunjungi satu kali dan total pengiriman tidak lebih dari C , dengan tujuan untuk meminimalkan total jarak tempuh semua kendaraan (Borgulya, 2008).



Gambar 2.1. Penyelesaian *Vehicle Routing Problem*
(Haksever et al. , 2000)

VRP merupakan salah satu kisah sukses dari *operational research*. Hal ini dapat dilihat banyaknya penelitian yang membahas tentang *Vehicle Routing Problem* dalam 50 tahun terakhir sejak tulisan Danzig dan Ramser keluar. Golden et al. (2008) dalam bukunya merangkum beberapa penelitian yang membahas VRP

yang dilakukan oleh peneliti-peneliti lain. Kumpulan penelitian yang dirangkum terdiri dari tiga bagian adalah survey dan gambaran tentang VRP, penerapan VRP, pengembangan model dan algoritma.

Perkembangan kasus distribusi di dunia nyata dengan berbagai macam karakteristik membuat banyaknya varian VRP yang merupakan pengembangan dari varian VRP yang sudah ada. Mulai dari VRP dengan *single objective* hingga *multi objective*. Hal ini yang membuat banyak peneliti yang merasa tertantang untuk menyelesaikan kasus VRP yang mulai beragam dan lebih *up to date* seperti yang dilakukan oleh Hemsch dan Irnich (2008) atau mengembangkan metode baru untuk mencari solusi yang lebih baik seperti Baldacci *et al.* (2008). Hemsch dan Irnich (2008) menyebutkan bahwa kasus distribusi pada dunia nyata memiliki kendala baru berupa *Inter-tour constraints*. *Inter-tour constraints* adalah kendala yang melihat bahwa terdapat banyak sifat dalam distribusi yang mempengaruhi solusi yang dihasilkan, seperti proses *sorting* pada depot, lama maksimum rute, dan terbatasnya kapasitas untuk memproses barang yang datang. Hemsch dan Irnich menggunakan *local-search algorithms* dalam menyelesaikan kasus tersebut. Sedangkan Baldacci *et al.* (2008) mengembangkan metode heuristik dalam menyelesaikan varian kasus *Heterogeneous Fleet VRP* (HFVRP). Tujuan dari penelitian Baldacci *et al.* adalah membandingkan hasil komputasi dan performansi algoritma heuristik miliknya dengan penelitian terdahulu.

Kasus VRP yang bersifat *multi objective* pernah dilakukan oleh Ueng (1999) yang mencoba membawa beban kerja supir taksi sebagai faktor yang diperhitungkan dan mengembangkan model VRP yang tidak hanya mencari jarak terpendek tetapi juga mempertimbangkan keseimbangan beban kerja yang dibebankan ke supir. Penelitian yang dilakukan Ueng memiliki kesamaan dengan penelitian ini, tetapi hal yang membedakan adalah definisi keseimbangan beban kerja. Ueng mendefinisikan keseimbangan beban kerja dengan meminimumkan nilai varian dari *load* yang harus diantarkan supir taksi sedangkan pada penelitian ini keseimbangan beban kerja didefinisikan dengan meminimasi rentang *load* kendaraan. Ueng mengusulkan algoritma heuristik dalam menyelesaikan kasus CVRPLB.

2.2. Particle Swarm Optimization

Particle Swarm Optimization (PSO) adalah suatu metode pencarian acak berbasis populasi. PSO didasarkan pada perilaku perpindahan individu dalam kawanan, seperti kawanan serangga, semut, rayap, lebah atau burung. Pertama kali PSO

diusulkan oleh Kennedy dan Eberhard pada tahun 1995. Secara konsep PSO adalah kemampuan intelegensia setiap partikel dalam sebuah kawanan dan kemampuan kawanan dalam menentukan posisi partikel (Nguyen *et al.*, 2010). Pada sebuah kawanan, jika ada salah satu individu (partikel) menemukan posisi yang lebih baik maka individu-individu lainnya dalam kawanan akan mendekat ke lokasi tersebut. PSO mempunyai algoritma yang dapat digunakan untuk berbagai masalah optimasi, yaitu :

- a. Tahap inisialisasi, yang merupakan tahap awal penentuan jumlah partikel, jumlah iterasi, posisi partikel dan kecepatan partikel.
- b. Proses penerjemahan posisi partikel ke dalam solusi yang akan dicari.
- c. Evaluasi posisi partikel terhadap fungsi objektifnya.
- d. Pembaharuan nilai *pbest* dengan nilai yang menghasilkan *fitness* terbaik dari partikel tersebut.
- e. Pembaharuan nilai *gbest* dengan nilai yang menghasilkan *fitness* terbaik dari semua partikel.
- f. Proses kembali ke langkah c jika batas iterasi belum terpenuhi.

Solusi yang dihasilkan algoritma PSO untuk kasus tertentu adalah sebuah posisi dimensional. Model disimulasikan dalam ruang dimensi dengan sejumlah iterasi hingga setiap iterasi akan semakin mendekat dengan posisi atau target yang akan dituju. Menurut Nguyen *et al.* (2010) setiap iterasi yang dihasilkan, kecepatan partikel dalam melakukan penyesuaian didasarkan pada 3 komponen yaitu: (1) kecepatan partikel yang merepresentasikan momentum partikel. (2) *pbest* (posisi yang sesuai dengan solusi terbaik dan dicapai oleh partikel). (3) *gbest* (posisi yang sesuai dengan solusi terbaik dan dicapai oleh semua partikel).

PSO merupakan metode optimasi yang dapat digunakan untuk memecahkan permasalahan optimasi. VRP merupakan suatu permasalahan optimasi dalam bidang distribusi, sehingga dapat diselesaikan dengan PSO. Jiang dan Li (2011) membandingkan algoritma PSO dan GA (*Genetic Algorithm*) dalam menyelesaikan kasus VRP. Dalam penelitiannya PSO menghasilkan hasil yang lebih baik dengan kecepatan operasi, ketahanan, dan hasil optimasi. Untuk kasus yang lebih rumit, PSO pernah digunakan untuk menyelesaikan kasus *Periodic Vehicle Routing Problem* (PVRP) dengan efektif yang merupakan gabungan dari CVRP, *Vehicle Routing Problem with Time Windows* (VRPTW), *Vehicle Routing Problem with Simultaneous Pickup and Delivery* (VRSPD), dan *Vehicle Routing Problem with Split Service* (VRPSS) (Moghaddam *et al.*, 2012).

2.3. Multi-objective Optimization Particle Swarm Optimization

Multi-objective Optimization adalah suatu proses yang secara simultan mengoptimalkan dua atau lebih fungsi tujuan yang saling bertentangan dengan kendala yang ada (Gutierrez, 2012). *Multi-objective Optimization* dalam kasus nyata banyak digunakan untuk menangani permasalahan yang harus memenuhi lebih dari satu fungsi tujuan. Model matematis untuk *Multi-objective Optimization* diformulasikan dengan :

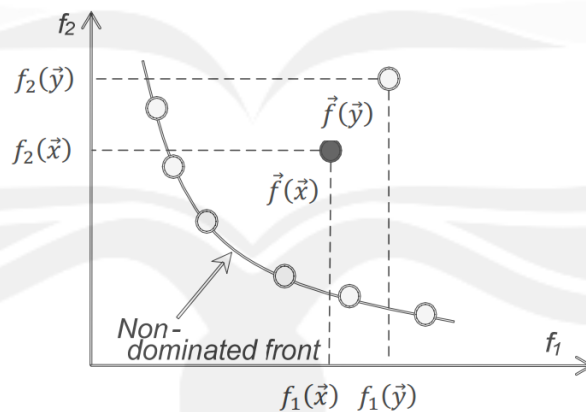
$$\text{minimize} \quad \vec{f}(\vec{x}) = [\vec{f}_1(\vec{x}), \vec{f}_2(\vec{x}), \dots, \vec{f}_k(\vec{x})] \quad (2.1)$$

Kendala :

$$g_i(\vec{x}) \leq 0 \quad i = 1, 2, \dots, m \quad (2.2)$$

$$h_i(\vec{x}) \leq 0 \quad i = 1, 2, \dots, i \quad (2.3)$$

Dengan ketentuan $\vec{x}$ adalah vector untuk variabel keputusan, $\vec{f}(\vec{x})$ adalah fungsi dari $\vec{x}$, k adalah jumlah fungsi tujuan yang akan diminimalkan, $g_i(\vec{x})$ dan $h_i(\vec{x})$ adalah kendala yang ada. *Multi-objective optimization* bertujuan untuk menemukan satu set solusi yang tidak saling mendominasi antar fungsi tujuan (*non-dominated front*). Gambar 2.2. adalah contoh *non-dominated pareto font* yang menunjukkan solusi *non-dominated* untuk $f(\vec{x})$ dan $f(\vec{y})$.

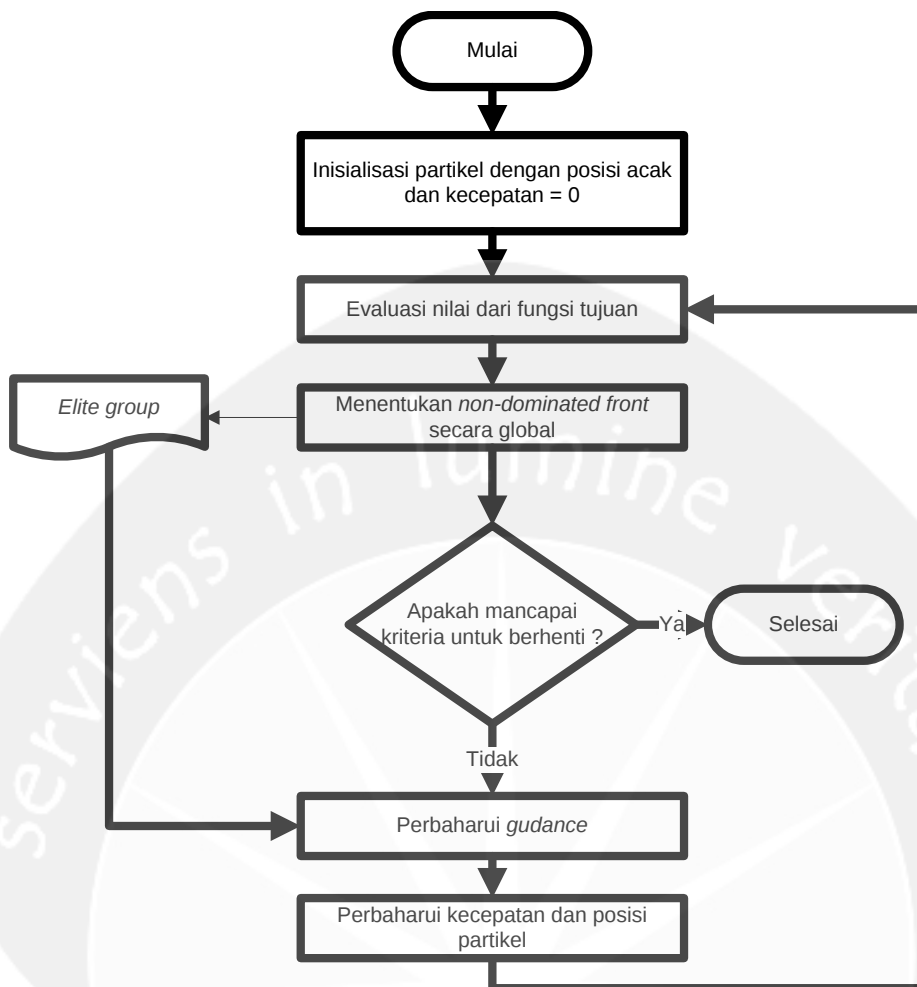


Gambar 2.2. Non-Dominated Front untuk Dua Fungsi Tujuan

Multi-objective PSO (MOPSO) adalah pengembangan algoritma PSO untuk menyelesaikan kasus *Multi-Objective Optimization*. Pada MOPSO, setelah setiap partikel memperbaharui posisi maka setiap partikel dievaluasi dengan fungsi tujuan. Partikel-partikel tersebut selanjutnya diproses dengan *non-dominated sorting procedure*. Prosedur ini partikel diseleksi untuk mencari *non-dominated particle* yang berikutnya dikelompokkan menjadi *elite group*. Setelah *elite group*

terbentuk maka dipilihlah kandidat di antara *elite group* yang akan menjadi *leader* (*guidance*) yang akan membawa kawanan ke area yang masih kosong untuk mendapatkan penyebaran pencarian yang lebih baik. Langkah-langkah yang disebutkan di atas digambarkan dalam sebuah *flowchart* pada Gambar 2.3.

MOPSO cukup populer di kalangan peneliti yang konsen dalam bidang optimasi. MOPSO banyak digunakan untuk menyelesaikan bermacam-macam permasalahan optimasi termasuk pula VRP. Norouzi *et al.* (2012) dalam penelitiannya membandingkan algoritma MOPSO dengan *Non-dominated Sorting Genetic Algorithm-II* (NSGA-II) dalam menyelesaikan *Open Vehicle Routing Problem* (OVRP). Kedua metode dibandingkan dari segi waktu dan ongkos perjalanan. Hasil perbandingan menunjukkan MOPSO lebih efisien dalam menyelesaikan OVRP dengan perhitungan ongkos perjalanan yang lebih baik. Pourrousta *et al.* (2002) mengimplementasi *ranking Method* dan MOPSO dalam mengintegrasikan rantai pasok dan membandingkan hasil dengan dengan algoritma NSGA-II. Masih banyak penelitian yang menggunakan algoritma MOPSO dalam menyelesaikan kasus *Routing Problem* seperti Alinaghian *et al.* (2012), Luo, *et al.* (2013) dan Liu *et al.* (2012).



Gambar 2.3. Flowchart MOPSO

2.4. Object Library for Evolutionary Techniques (ET-Lib) versi 1.0

ET-Lib merupakan perangkat umum yang dikembangkan oleh Nguyen *et al.* (2010) dan bekerja berdasarkan variasi teknik evolutioner. ET-Lib bertujuan untuk menyediakan peneliti dan pelajar suatu perangkat yang mampu menyelesaikan berbagai jenis masalah optimasi. ET-Lib versi 1.0 berisi Algoritma *Particle Swarm Optimization with Multiple Social Learning Term* (GLNPSO). Algoritma GLNPSO ditulis dalam bahasa C# sebagai sebuah *library*. *Library* tersebut terdiri dari *class* dan *method* yang digunakan untuk menjalankan algoritma PSO. GLNPSO merupakan pengembangan dari algoritma PSO dengan menambahkan 2 struktur sosial yaitu *local best* (lbest), dan *near neighbor best* (nbest). *Local best* menerima nilai *fitness* terbaik dari subgroup dan tiap partikel dapat memperbaharui kecepatan berdasarkan performansi terbaik dari *neighbor* dalam populasi dari partikel tersebut. *Near neighbor best* berisi nilai maksimum *Fitness Distance Ratio*

(FDR) dari semua partikel yang ada. Berikut algoritma GLNPSO yang telah disesuaikan untuk kasus CVRP

Algoritma 1. Algoritma GLNPSO untuk CVRP

1. Pada iterasi pertama ($t = 1$) inialisasi sejumlah l partikel sebagai populasi dan munculkan partikel ke- i dengan posisi acak X_i dalam dalam jangkauan $[X^{min}, X^{max}]$, dengan kecepatan $V_i = 0$ dan $pbest$ $P_i = X_i$ untuk $i = 1 \dots l$.
2. Untuk $i = 1 \dots l$, setiap posisi X_i partikel diterjemahkan menjadi sebuah rute R_i
3. Untuk $i = 1 \dots l$, setiap R_i dievaluasi berdasarkan fungsi tujuan yang ada dan jadikan menjadi *fitness value* dari X_i yang disimbolkan dengan $\varphi(X_i)$.
4. Perbaharui nilai dari $pbest$: $P_i = X_i$, jika $\varphi(X_i) < \varphi(P_i)$, ($i = 1 \dots l$)
5. Perbaharui nilai dari $gbest$: $P_g = P_i$, jika $\varphi(P_i) < \varphi(P_g)$, ($i = 1 \dots l$)
6. Perbaharui $lbest$: untuk $i = 1 \dots l$, di antara semua $pbest$ pada K neighbors dari partikel ke- i , pilihlah salah satu $pbest$ yang memiliki *fitness value* paling kecil menjadi P_i^L .
7. Memunculkan $nbest$: untuk $i = 1 \dots l$ dan $d = 1 \dots D$ tetapkan $P_{id}^N = P_{jd}$ yang memaksimalkan *fitness-distance-ratio* (FDR) untuk $j = 1 \dots I$, dimana FDR dirumuskan dengan :

$$FDR = \frac{\varphi(X_i) - \varphi(P_i)}{|X_{id} - P_{jd}|} \quad \text{dimana } i \neq j \quad (2.4)$$

8. Perbaharui kecepatan dan posisi untuk setiap partikel :

$$w(t) = w(T) + \frac{t-T}{1-T} [w(1) - w(T)] \quad (2.5)$$

$$v_{id}(t+1) = w(t)v_{id}(t) + c_p u(P_{id} - X_{id}(t)) + C_g u(P_{gd} - X_{id}(t)) + C_l u(P_{lid} - X_{id}(t)) + C_n u(P_{nid} - X_{id}(t)) \quad (2.6)$$

$$x_{id}(t+1) = X_{id}(t) + v_{id}(t+1) \quad (2.7)$$

Jika $x_{id}(t+1) > X^{max}$,

$$x_{id}(t+1) = X^{max} \quad (2.8)$$

$$v_{id}(t+1) = 0 \quad (2.9)$$

Jika $x_{id}(t+1) < X^{min}$,

$$x_{id}(t+1) = X^{min} \quad (2.10)$$

$$v_{id}(t+1) = 0 \quad (2.11)$$

9. Jika kriteria berhenti sudah terpenuhi ($t = T$), lanjut ke langkah 10. Namun jika belum maka kembali ke langkah 2 dengan memperbaharui nilai t ($t = t + 1$)
10. *Decode* nilai P_g dan tetapkan sebagai rute terbaik yang dihasilkan R^* .

Keterangan:

t	Iterasi ; $t = 1, \dots, T$
i	Partikel ; $i = 1, \dots, I$
d	Dimensi ; $d = 1, \dots, D$
u	Bilangan random <i>uniform</i> pada interval $[0,1]$
$w(t)$	Beban inersia pada iterasi ke- t
$v_{id}(t)$	Kecepatan partikel ke- i pada dimensi ke- d di iterasi ke- t
$x_{id}(t)$	Posisi partikel ke- i pada dimensi ke- d di iterasi ke- t
p_{id}	Posisi <i>personal best</i> (pbest) dari partikel ke- i pada dimensi ke- d
p_{gd}	Posisi <i>global best</i> (gbest) pada dimensi ke- d
p_{id}^L	Posisi <i>local best</i> (lbest) dari partikel ke- i pada dimensi ke- d
p_{id}^N	Posisi <i>neighbor best</i> (nbest) dari partikel ke- i pada dimensi ke- d
c_p	Konstanta percepatan untuk pbest
c_g	Konstanta percepatan untuk gbest
c_l	Konstanta percepatan untuk lbest
c_n	Konstanta percepatan untuk nbest
X_i	Vektor posisi dari partikel ke- i ; $[x_{i1}, x_{i2}, \dots, x_{iD}]$
V_i	Vektor kecepatan dari partikel ke- i ; $[v_{i1}, v_{i2}, \dots, v_{iD}]$
P_i	Vektor posisi pbest dari partikel ke- i ; $[p_{i1}, p_{i2}, \dots, p_{iD}]$
R_i	Sekumpulan rute yang didapatkan dari partikel ke- i
$\varphi(X_i)$	<i>Fitness value</i> dari X_i
X^{max}	Posisi maksimum partikel
X^{min}	Posisi minimum partikel
FDR	<i>Fitness-Distance-Ratio</i>

GLNPSO telah sukses diaplikasikan dalam menyelesaikan banyak *NP-hard combinatorial problem*. Contohnya seperti masalah penjadwalan *job shop*, *vehicle routing problem*, *multicommodity distribution network design problem* dan lain-lain. Ai (2008) melakukan penelitian dengan menggunakan algoritma GLNPSO untuk menyelesaikan kasus *generalized vehicle routing problem* (GVRP) yang merupakan generalisasi dari 4 varian VRP yaitu CVRP, VRPTW, HVRP, dan VRPSPD.

Penelitian ini menggunakan bantuan *library* M3PSO (*Multi-strategy Multi-learning Term Multi-Objective Particle Swarm Optimization*) milik ET-Lib. *Library* M3PSO berisi *class* dan *method* yang digunakan untuk menjalankan algoritma MOPSO. Penggunaan *library* MOPSO membuat penyusunan program menjadi lebih mudah, dengan menyesuaikan algoritma MOPSO dengan CVRPLB. Penyesuaian

algoritma dalam penyusunan program juga didasarkan pada program yang dibuat oleh Ai (2008). Hal ini membuat proses penyusunan program dilakukan dengan sedikit penyesuaian penulisan *coding* untuk CVRPLB.

