Implementasi algoritma *harmony search* (HS) pada *mix fleet vehicle routing problem with split delivery* (MFVRPSD)

Erlina Tri Astuti[1], Mimiep Setyowati Madja[2], dan Mohamad Yasin[3]
Universitas Negeri Malang
E-mail: erlinatri23@gmail.com

ABSTRAK: Salah satu varian dari VRP adalah *Mix Fleet Vehicle Routing Problem with Split Delivery (MFVRPSD)*, dimana kendaraan yang dipakai bisa memiliki kapasitas yang berbeda-beda dan *customer* dapat dikunjungi lebih dari satu kali agar mendapatkan rute yang minimum. Pada MFVRPSD permintaan *customer* tidak boleh melebihi kapasitas kendaraan. MFVRPSD bertujuan untuk menentukan sejumlah rute yang memiliki jarak tempuh minimum. Semua rute berawal dan berakhir di depot yaitu 0. Total permintaan dari sebarang rute kendaraan tidak boleh melebihi kapasitas kendaraan. Dan kapasitas dari masing-masing kendaraan yang digunakan untuk melayani customer bisa berbeda-beda sesuai yang tersedia di tempat pengiriman atau produsen. Algoritma *Harmony Search* dapat digunakan untuk menyelesaikan permasalahan MFVRPSD dengan tujuan untuk memperoleh solusi berupa rute yang optimum tanpa melanggar kendala kapasitas dan jarak. Algoritma *Harmony Search* memiliki 5 langkah utama yaitu inisialisasi parameter, inisialisasi harmony memory (HM), membangkitkan rute sementara, meng-update HM rute sementara, dan mengecek kriteria pemberhentian. Agar mudah dalam menyelesaikan permasalahan MFVRPSD dengan menggunakan Algoritma *Harmony Search*, maka akan direpresentasikan dalam program komputer dengan menggunakan *Borland Delphi 7.0*. Program dimulai dengan input data, kemudian data diproses dengan menggunakan algoritma *Harmony Search* dan output yang dihasilkan berupa rute yang optimum serta visualisasi graph hasilnya.

Kata kunci: Varian VRP, MFVRPSD, Algoritma *Harmony Search*

ABSTRACT: One of the variants of the VRP is Mix Fleet Vehicle Routing Problem with Split Delivery (MFVRPSD), where the vehicle is used may have different capacities and the customer can be visited more than once in order to get the minimum route. At MFVRPSD, customer demand may not exceed the capacity of the vehicle. MFVRPSD aims to determine the number of routes that have the minimum mileage. All routes begin and end at the depot 0. Total demand of any vehicle route should not exceed the capacity of the vehicle. And the capacity of each vehicle that is used to serve the customer

1. Erlina Tri Astuti adalah mahasiswa Jurusan Matematika FMIPA Universitas Negeri Malang
2. Mimiep Setyowati Madja adalah dosen Jurusan Matematika FMIPA Universitas Negeri Malang
3. Mohamad Yasin adalah dosen Jurusan Matematika FMIPA Universitas Negeri Malang

can vary according to the vehicle are available at the manufacturer. Harmony Search Algorithm can be used to solve the problems MFVRPSD in order to obtain solutions such as an optimum route without violating the capacity and the distance constraints. Harmony Search Algorithms has four main steps, ie harmony initialization memory (HM), generate the temporary route, updating HM temporary route, and check the criteria for termination. To make it easier in resolving problem of MFVRPSD using Harmony Search Algorithm, it will be represented in a computer program using Borland Delphi 7.0. The program begins with the input of the data, then the data is processed using an Harmony Search algorithm and generate output optimum route, as well as the visualization of the results graph.

Key Word: VRP variants, MFVRPSD, *Harmony Search* Algorithm

Distribusi merupakan proses pengiriman barang atau jasa yang biasanya digunakan untuk melayani beberapa *customer*-nya oleh sebuah perusahaan atau jasa pengiriman. Ketika proses distribusi berlangsung, produsen harus memperhatikan beberapa kendala yang bisa saja terjadi, misalnya saja kapasitas masing-masing kendaraan, waktu pengiriman, permintaan customer, dan kecepatan dari masing-masing kendaraan. Dan salah satu cabang dari matematika yang dapat digunakan untuk menyelesaikan permasalahan distribusi tersebut adalah teori graph. Konsep pada teori graph yang digunakan adalah *Vehicle Routing Problem* (VRP). VRP merupakan permasalahan untuk mencari rute optimum yang akan digunakan oleh beberapa kendaraan yang akan melayani beberapa customer dimana kendaraan akan berawal dan berakhir di depot (perusahaan/produsen) serta kendaraan mengunjungi setiap customer tepat satu kali.

Salah satu varian dari VRP adalah *Heterogeneous* VRP (*Mix Fleet* VRP). *Mix Fleet Vehicle Routing Problem* (MFVRP) sendiri merupakan salah satu varian dari VRP dimana pada saat terjadi pengiriman barang, kendaraan yang digunakan memiliki kapasitas yang berbeda-beda. Ada juga varian lain dari VRP yaitu *Split Delivery Vehicle Routing Problem* (SDVRP) yang merupakan permasalahan pada VRP dimana proses pendistribusian berawal dan berakhir di depot dan tiap *customer* boleh dikunjungi lebih dari satu kali oleh lebih dari satu kendaraan tanpa melanggar batasan kapasitas, yaitu dalam satu rute jumlah permintaan *customer* tidak melebihi kapasitas angkut kendaraan. Algoritma yang dapat digunakan untuk menyelesaikan permasalahan VRP salah satunya adalah algoritma *Harmony Search*. Berdasarkan latar belakang yang dikemukakan di atas, maka akan dikaji algoritma *Harmony Search* dalam menyelesaikan *Mix Fleet Vehicle Routing Problem with Split Delivery*.

HASIL YANG DIHARAPKAN

1. Mengetahui langkah- langkah dan implementasi algoritma *Harmony Search* dalam menyelesaiakan permasalahan MFVRPSD.
2. Mengetahui implementasi algoritma *Harmony Search* dalam menyelesaikan permasalahan MFVRPSD dengan menggunakan bahasa pemrograman *Borland Delphi 7. 0* dan contoh aplikasinya.

HASIL DAN PEMBAHASAN

1. *Mix Fleet Vehicle Routing Problem with Split Delivery*

MFVRPSD didefinisikan sebagai suatu graph $G = (N, A)$ dengan $N = \{1,2, \dots, n\}$ adalah himpunan titik dan $A = \{(i,j) | \, i,j \in N, i \neq j\}$, dimana suatu titik mewakili sebuah depot dan titik-titik yang lain mewakili beberapa *customer*. Pada MFVRPSD, suatu himpunan n *customer* dilayani oleh k jenis kendaraan, dengan $k = 1,2, \dots, K$ dan banyak kendaraan dengan jenis k dinotasikan dengan m_k. Kapasitas masing-masing kendaraan dinotasikan dengan Q_k. Setiap *customer* memiliki permintaan sejumlah d_i dengan $i = 1,2, \dots, n$, dimana permintaan *customer* sama besar atau lebih kecil dari kapasitas kendaraan.

MFVRPSD bertujuan untuk menentukan sejumlah rute yang memiliki jarak tempuh minimum. Semua rute berawal dan berakhir di depot yaitu 0. Total permintaan dari sebarang rute kendaraan tidak boleh melebihi kapasitas kendaraan. Dan kapasitas dari masing-masing kendaraan yang digunakan untuk melayani *customer* bisa berbeda-beda sesuai yang tersedia di tempat pengiriman atau produsen. Beberapa hal yang perlu diperhatikan dalam permasalahan MFVRPSD adalah sebagai berikut:

a. Setiap *customer* terhubung dengan depot dan bisa dikunjungi lebih dari satu kendaraan.
b. Rute kendaraan berawal dan berakhir di depot.
c. Memenuhi kendala kapasitas.
d. Kecepatan rata-rata kendaraan diasumsikan sama(konstan) untuk tiap kendaraan.
e. Jumlah kendaraan yang dibutuhkan depot untuk mendistribusikan barang sesuai dengan jumlah rute yang terbentuk dan diasumsikan selalu tersedia dengan kapasitas yang berbeda-beda tergantung jenis kendaraan.
f. Komoditi barang yang didistribusikan sejenis.

2. Algoritma *Harmony Search(HS)* pada MFVRPSD

Algoritma HS sendiri dikembangkan oleh Zong Woo Geem pada tahun 2001 untuk menyelesaikan permasalahan optimasi dan telah diaplikasikan dalam beberapa permasalahan dimana salah satunya adalah pada permasalahan optimasi rute kendaraan. Algoritma HS adalah salah satu algoritma metaheuristik yang terinspirasi dari improvisasi musik pengamatan yang dilakukan oleh musisi pada sebuah kelompok paduan musik untuk mencari keadaan harmoni yang lebih baik berdasarkan perkiraan estetika. Usaha untuk mencari harmoni terbaik dalam kelompok paduan musik tersebut dapat dianalogikan sebagai usaha untuk mencari rute optimal dari suatu permasalahan MFVRPSD.

Terdapat 6 langkah algoritma HS yang digunakan untuk menyelesaikan MFVRPSD, antara lain:

2.1.Identifikasi permasalahan.

Mengidentikasi permasalahan yang akan diselesaikan yaitu MFVRPSD dengan meminimumkan total jarak tempuh $h(x_i)$ dengan $x_i \in X_i$, $i = 1,2, \dots, N$; $BB \leq X_i \leq BA$

dimana $h(x_i)$ adalah total jarak tempuh
x_i adalah *customer* ke-i
X_i adalah himpunan *customer*
N adalah jumlah *customer*
BB adalah batas bawah *customer*
BA adalah batas atas *customer*

2.2.Inisialisasi parameter.

Terdapat beberapa parameter yang digunakan pada algoritma HS agar *Harmony Memory*(HM) dapat digunakan dengan efektif, antara lain:

a. *Harmony Memory Search* (HMS)
HMS adalah solusi atau banyaknya rute sementara yang terbentuk dan disimpan pada HM.
b. *Harmony Memory Consideration Rate* (HMCR)
HMCR bernilai $0 \leq HMCR \leq 1$. Umumnya berkisar 0,7 sampai 0,95 (Maftuh, 2010).
c. *Bandwidth* (bw)
Bandwidth (bw) MFVRP adalah sebarang bilangan bulat positif. *Bandwidth* berguna untuk mengubah frekuensi nada atau menyesuaikan *customer*. Persamaan berikut ini adalah rumus penyesuaian *customer* :
$NP = x_{old} + bw \times \varepsilon$
dimana NP adalah nilai penyesuaian *customer* baru
x_{old} adalah *customer* yang tersimpan pada *hamony memory* rute sementara.
bw adalah sebarang bilangan bulat positif
ε adalah bilangan random dengan interval [-1,1]
d. *Pitch Adjustment Rate* (PAR)
PAR bernilai $0 \leq PAR \leq 1$. Umumnya berkisar antara 0,1 sampai 0,5 (Maftuh, 2010).
e. Kriteria pemberhentian (NI)
Kriteria pemberhentian tercapai jika iterasi yang dilakukan sama dengan jumlah iterasi yang diinputkan.

2.3.Inisialisasi *Harmony Memory* (HM).

Pada langkah ini, suatu himpunan solusi dari ukuran HMS dibangkitkan untuk membangun HM. HM digambarkan sebagai suatu matriks 2 dimensi. Baris menunjukkan suatu himpunan solusi atau disebut vector solusi X^i, sedangkan kolom menunjukkan variabel keputusan untuk tiap solusi. Setiap solusi X^i dapat dilihat sebagai salah satu susunan urutan. Berikut ini adalah contoh matriks HM :

$$HM = \begin{pmatrix} x_1^1 & x_2^1 & \cdots & x_{N-1}^1 & x_N^1 \\ x_1^2 & x_2^2 & & x_{N-1}^2 & x_N^2 \\ \vdots & & \ddots & \vdots & \\ x_1^{HMS-1} & x_2^{HMS-1} & & x_{N-1}^{HMS-1} & x_N^{HMS-1} \\ x_1^{HMS} & x_1^{HMS} & \cdots & x_{N-1}^{HMS} & x_N^{HMS} \end{pmatrix}$$

Keterangan :

Matriks di atas memiliki jumlah baris sebanyak HMS. Baris pertama hingga baris ke-HMS masing-masing terdiri dari semua *customer* yang terbentuk secara acak.

2.4. Membangkitkan *Harmony* Baru.

Proses pembangkitan rute sementara baru, diawali dengan merandom bilangan bulat dari 1 sampai banyaknya *customer* yang akan dilayani. Kemudian, menyesuaikannya dengan nilai HMCR dan PAR seperti berikut:
Misalnya saja dibangkitkan bilangan random r_1, jika:

- $r_1 \geq HMCR$ maka akan dipilih *customer* yang baru secara random.
  $CB(i) = rand[BB, BA]$
  dimana $CB(i)$ adalah *customer* $ke-i$
  BB adalah batas bawah *customer*
  BA adalah batas atas *customer*
- $r_1 < HMCR$, maka akan dibangkitkan lagi bilangan random r_2, selanjutnya jika:
  - $r_2 \geq PAR$, maka *customer* ke$-i$ akan diambil secara random dari HM dengan $CB(i) \in \{x_i^1, x_i^2, \dots, x_i^{HMS-1}, x_i^{HMS}\}$.
    Dimana CBS(i) = *customer* baru sementara ke i.
  - $r_2 < PAR$, maka terdapat penyesuaian terhadap *customer* ke$-i$ yang akan diambil dari HM, yaitu sebagai berikut:
    1) Lakukan random *customer* dengan $CBS(i) \in \{x_i^1, x_i^2, \dots, x_i^{HMS}\}$.
    2) Hitung nilai penyesuaian *customer* baru dengan rumus sebagai berikut:
       $$NP = CBS(i) + bw \times \varepsilon$$
       dimana NP adalah nilai penyesuaian *customer* baru.
       bw adalah bilangan bulat positif
       ε adalah bilangan random dengan interval [-1,1]

Kemudian, setelah dilakukan proses penyesuaian maka akan dilakukan proses pengecekan terhadap batas atas dan batas bawah dari *customer*. Mula-mula, cek apakah $NP < BB$. Jika ya, maka $CB(i) = BB$. Jika tidak, cek apakah $NP > BA$. Jika ya, maka $CB(i) = BA$. Jika tidak, maka $CB(i) = CBS(i)$.

2.5. *Harmony Memory Update* (Meng-update HM).

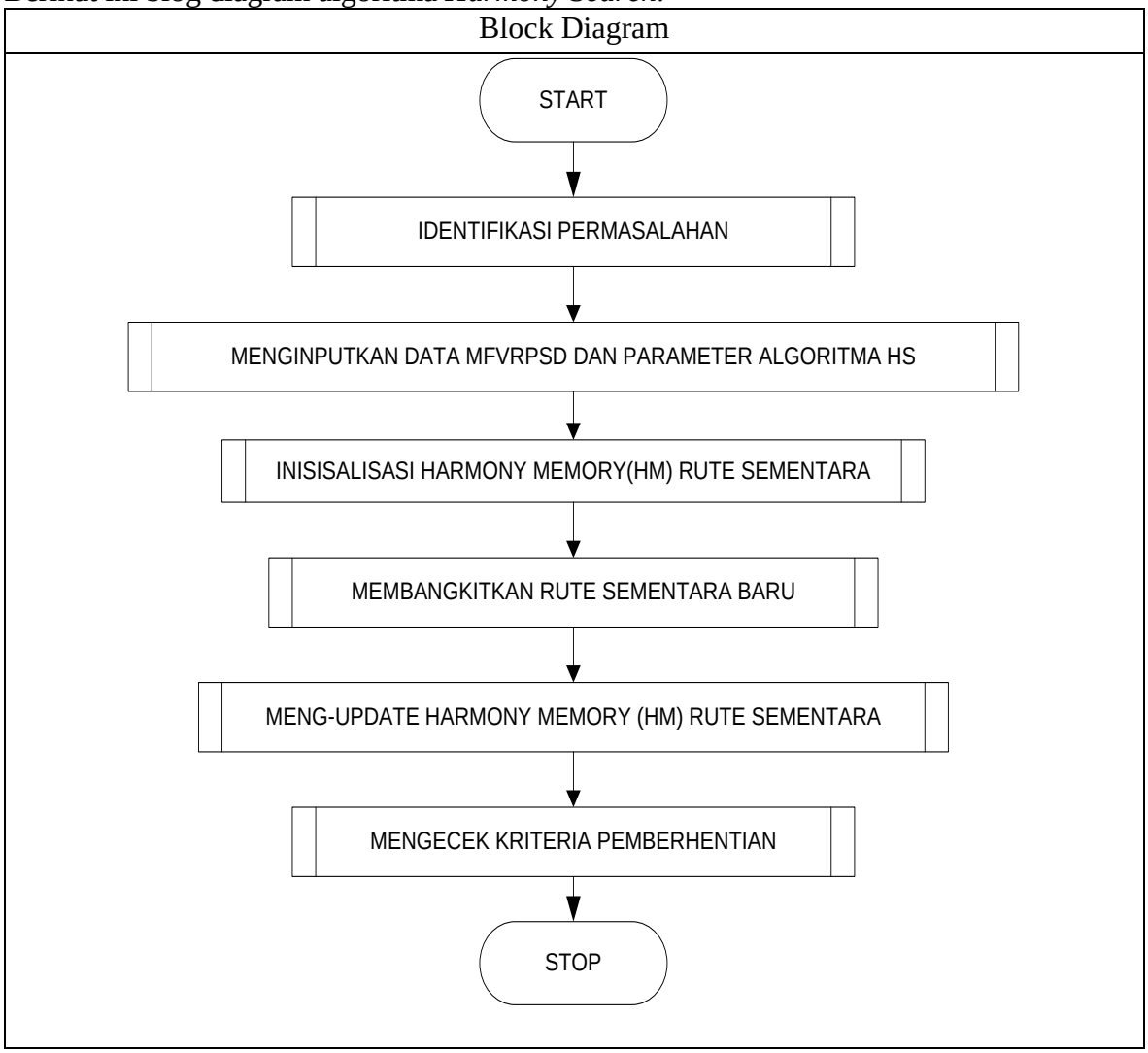
Jika rute sementara yang baru menghasilkan total jarak tempuh yang lebih pendek dari rute sementara dengan total jarak tempuh terpanjang pada HM, maka rute sementara yang baru akan dimasukkan ke dalam HM menggantikan rute sementara dengan total jarak tempuh terpanjang. Jika total jarak tempuh rute

sementara yang baru lebih panjang daripada rute sementara dengan total jarak tempuh terpanjang pada HM, maka HM tidak berubah.

2.6. Mengulangi langkah 4 dan 5 sampai kriteria pemberhentian tercapai dan mengecek kriteria pemberhentian.

Apabila kriteria pemberhentian telah tercapai, yaitu jika iterasi yang dilakukan sama dengan jumlah iterasi yang diinputkan maka iterasi dihentikan. Apabila belum tercapai maka kembali ke langkah 4.

Berikut ini blog diagram algoritma *Harmony Search*.



Gambar 1. Blog Diagram Algoritma *Harmony Search*

Setelah membuat rancangan *flowchart* dalam bentuk blog diagram, maka selanjutnya merancang program yang akan dibuat. Program dimulai dengan input data. Input yang pertama adalah input titik yang mewakili depot dan *customer* dengan menggunakan mouse (Satyananda: 2012). Sisi pada pada graph diwakili oleh matriks bobot. Data jarak, permintaan *customer* dan yang lainnya diinputkan menggunakan *keyboard*. Kemudian data diproses dengan menggunakan algoritma *Harmony Search*

dalam bahasa Pemrograman *Borland Delphi 7.0* dan output yang dihasilkan berupa rute yang optimum. Rute tersebut ditampilkan dalam memo dan graph hasil yang menunjukkan rute yang terbentuk. Pada program juga dilengkapi dengan button Buka, Simpan, Proses, Keluar dan Reset.

3. Contoh Penerapan *Mix Fleet Vehicle Routing Problem with Split Delivery* dengan Algoritma *Harmony Search*

Contoh ini diambil dari skripsi yang menyelesaikan *Mix Fleet Vehicle Routing Problem* (MFVRP) menggunakan Algoritma *Sweep* (Maulidiyah, 2011) dengan mengambil tujuh titik sebagai *customer* yang akan dilayani. Adapun daftar permintaan tiap cutomer seperti pada tabel berikut:

Tabel 3.1. Tabel permintaan *customer*

Customer	**1**	**2**	**3**	**4**	**5**	**6**	**7**
d_i	40	32	56	22	19	48	30

Daftar jarak dari depot ke costumer serta jarak *customer* satu ke *customer* lainnya ditampilkan pada Tabel 3.2 berikut:

Tabel 3.2. Tabel jarak

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	24	41	32	11	10	6	21
1		0	14	24	17	30	28	27
2			0	12	26	34	38	25
3				0	24	16	15	9
4					0	5	4	7
5						0	3	8
6							0	13
7								0

Sedangkan daftar kapasitas dari masing-masing kendaraan yang tersedia adalah sebagai berikut:

Tabel 3.3. Tabel kapasitas

Kendaraan (k)	**Qk**
1	75
2	72
3	150

Selanjutnya, akan dicari rute minimum dengan menggunakan Algoritma *Harmony Search*.

4. Penyelesaian menggunakan Algoritma *Harmony Search.*

4.1.Identifikasi Permasalahan

Urutkan permintaan dari yang terbesar ke yang terkecil, sehingga didapat daftar kendaraan baru, yaitu:

Tabel 3.4 Tabel Kapasitas Kendaraan (Urut)

Kendaraan	Kapasitas Kendaraan
Q_3	150
Q_1	75
Q_2	72

4.2.Identifikasi Parameter untuk HS

Parameter untuk algoritma HS akan dipilih secara random dengan menggunakan *Ms Excel*, diperoleh:

- HMCR dipilih antara 0,7 – 0,95. HMCR = 0,82
- PAR dipilih antara 0,1 – 0,5. PAR = 0,42.
- HMS = 3.
- bw = 10
- Kriteria pemberhentian dipilih 1 iterasi, NI = 1.

4.3.Inisialisasi HM Rute Sementara

HM dibangun sebanyak HMS yang telah ditentukan. HM merupakan matriks yang berisi sekumpulan rute sementara yang dibangun secara acak. Berikut merupakan HM rute sementara yang dibangkitkan sebanyak HMS.

$$HM = \left(\begin{array}{ccccccc|l} 2 & 3 & 1 & 4 & 6 & 7 & 5 & f(x_i) = 156 \\ 1 & 4 & 2 & 6 & 3 & 5 & 7 & f(x_i) = 271 \\ 5 & 1 & 3 & 4 & 2 & 7 & 6 & f(x_i) = 252 \end{array}\right)$$

dengan $X_1 = [2\ 3\ 1\ 4\ 6\ 7\ 5]$

$X_2 = [1\ 4\ 2\ 6\ 3\ 5\ 7]$

$X_3 = [5\ 1\ 3\ 4\ 2\ 7\ 6]$

Kemudian, dilanjutkan dengan mencari total jarak tempuh dari masing-masing rute sementara X_1, X_2, dan X_3.

Pada rute sementara pertama diperoleh:

- Rute(1) = [0,2,3,1,4,0] dengan jarak yang ditempuh kendaraan 3 adalah 105 km.
- Rute(2) = [0,6,0] dengan total jarak yang ditempuh kendaraan 1 adalah 12 km.
- Rute(3) = [0,7,5,0] dengan total jarak tempuh kendaan 2 adalah 39 km.

Dan didapat total jarak tempuhnya adalah 156 km.

Pada rute sementara kedua diperoleh:

- Rute(1) = [0,1,4,2,6,3,0] dengan jarak yang ditempuh kendaraan 3 adalah 152 km.
- Rute(2) = [0,3,5,7,0] dengan total jarak yang ditempuh kendaraan 1 adalah 77 km.

- Rute(3) = [0,7,0] dengan total jarak tempuh kendaan 2 adalah 42 km.

Dan didapat total jarak tempuhnya adalah 271 km.

Selanjutnya, pada rute sementara ketiga diperoleh:

- Rute(1) = [0,5,1,3,4,2,0] dengan jarak yang ditempuh kendaraan 3 adalah 155 km.
- Rute(2) = [0,2,7,6,0] dengan total jarak yang ditempuh kendaraan 1 adalah 85 km.
- Rute(3) = [0, 6,0] dengan total jarak tempuh kendaan 2 adalah 12 km.

Dan didapat total jarak tempuhnya adalah 252 km.

4.4. Membangkitkan *Harmony* Baru untuk Rute Sementara

Pada langkah 4 ini, akan dibangkitkan harmoni baru untuk rute sementara, yaitu:

$$X' = [x_1' \;\; x_2' \;\; x_2' \;\; x_3' \;\; x_4' \;\; x_5' \;\; x_6' \;\; x_7']$$

Dan didapat rute sementara sabagai berikut:

$X' = [3 \;\; 1 \;\; 7 \;\; 2 \;\; 4 \;\; 6 \;\; 7]$.

4.5. Meng-*update* HM rute sementara

Total jarak yang ditempuh untuk *harmony* baru rute sementara adalah sebagai berikut:

- Rute(1) = [0,3,1,7,2,0] dengan jarak yang ditempuh kendaraan 3 adalah 149 km.
- Rute(2) = [0,2,4,6,0] dengan total jarak yang ditempuh kendaraan 1 adalah 77 km.
- Rute(3) = [0,6,5,0] dengan total jarak tempuh kendaan 2 adalah 19 km.

Dan didapat total jarak tempuhnya adalah 245 km.

Karena rute sementara yang baru menghasilkan total jarak tempuh yang lebih pendek dari rute sementara dengan total jarak tempuh terpanjang pada HM, maka rute sementara yang baru akan dimasukkan ke dalam HM menggantikan rute sementara dengan total jarak tempuh terpanjang. Sehingga HM menjadi:

$$HM = \left(\begin{array}{ccccccc|l} 2 & 3 & 1 & 4 & 6 & 7 & 5 & f(x_i) = 156 \\ 3 & 1 & 7 & 2 & 4 & 6 & 5 & f(x_i) = 245 \\ 5 & 1 & 3 & 4 & 2 & 7 & 6 & f(x_i) = 252 \end{array}\right)$$

4.6. Mengecek Kriteria Pemberhentian.

Karena jumlah iterasi NI=1, maka jumlah iterasi maksimum terpenuhi, sehingga iterasi berhenti.

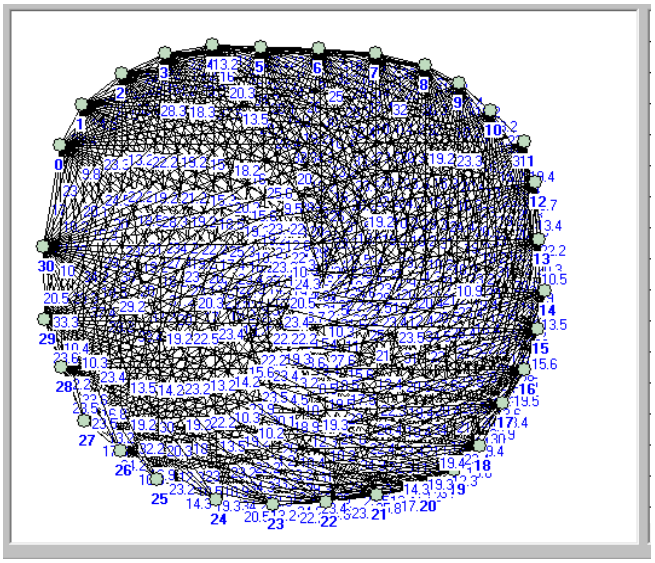
Sehingga, rute minimum yang didapat adalah :

- Rute(1) = [0,2,3,1,4,0] dengan jarak yang ditempuh kendaraan 3 adalah 105 km.
- Rute(2) = [0,6,0] dengan total jarak yang ditempuh kendaraan 1 adalah 12 km.
- Rute(3) = [0,7,5,0] dengan total jarak tempuh kendaan 2 adalah 39 km.

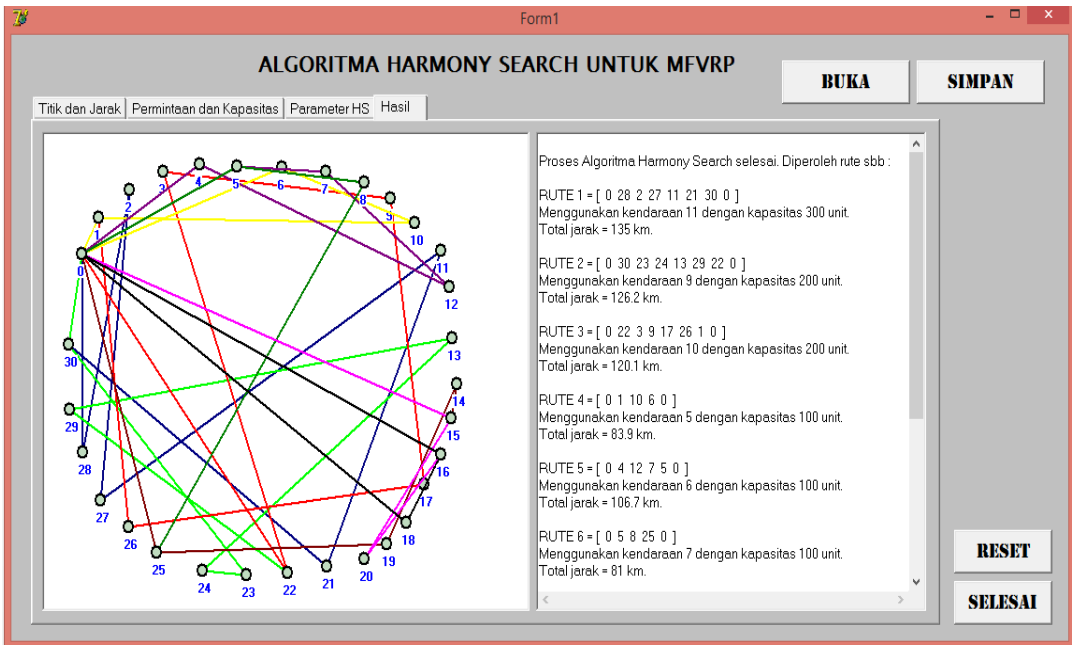
Dan didapat total jarak tempuhnya adalah 156 km.

5. Penyelesaian menggunakan program

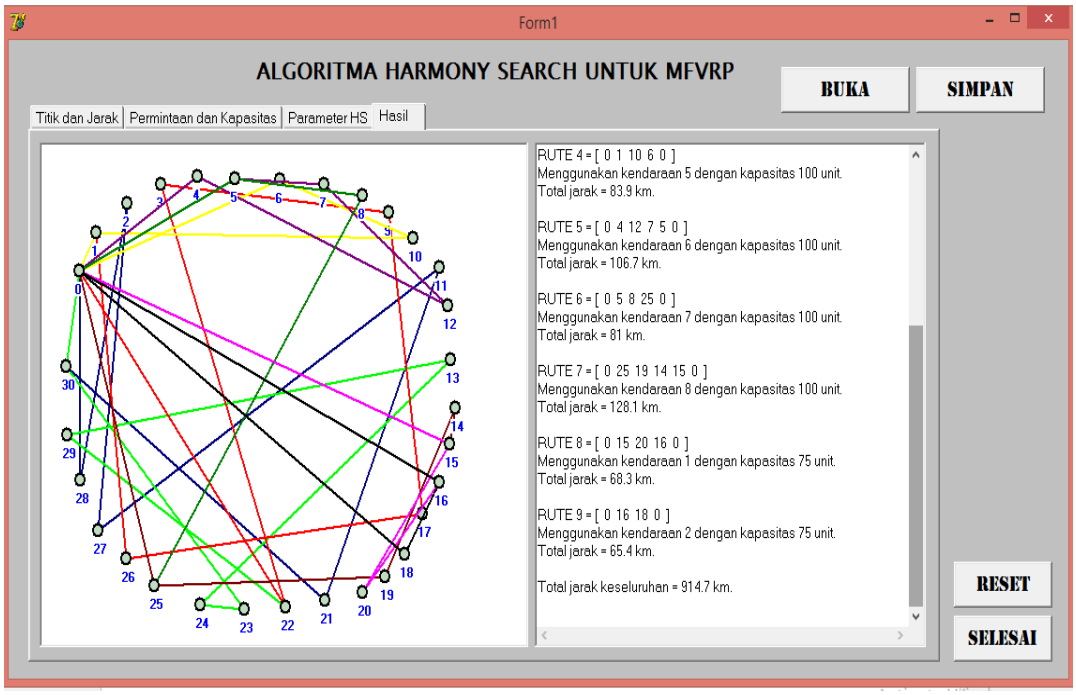
Data yang akan ditampilkan pada simulasi program ini adalah data dengan 30 titik. Hasil yang diperoleh adalah sebagai berikut.



Gambar 2. Graph awal



Gambar 3. Gambar Output 30 customer (1)



Gambar 4. Gambar Output 30 customer (2)

PENUTUP

Pada artikel ini telah dibahas permasalahan MFVRPSD, dimana dalam menyelesaikan permasalahan tersebut menggunakan Algoritma *Harmony Search* dan program yang dibuat dengan menggunakan bahasa pemrograman *Borland Delphi 7.0*. Berdasarkan hasil, pembahasan, contoh penerapan, dan uji coba menggunakan 8 titik dengan 27 kemungkinan parameter yang berbeda-beda yang telah dikerjakan dapat disimpulkan bahwa:

Program telah diuji coba dengan menggunakan parameter NI, HMS, dan bw dengan nilai 5, 50, dan 100 dengan 27 kemungkinan yang berbeda-beda. Dari uji coba yang telah dilakukan, parameter bw tidak begitu mempengaruhi hasil yang diperoleh yaitu walaupun bw berbeda-beda, rute yang didapat akan tetap pada kisaran tertentu. Parameter HMS cukup mempengaruhi hasil yang diperoleh, yaitu semakin tinggi HMS maka kemungkinan hasil yang didapat akan lebih optimum. Sedangkan parameter NI juga cukup mempengaruhi hasil yang diperoleh, yaitu semakin tinggi NI maka akan didapat rute yang lebih optimum. Program yang telah dibuat juga telah diuji coba dengan 30 dan 50 titik tanpa mengalami masalah *(error)* sehingga program tersebut dapat digunakan untuk menyelesaikan MFVRPSD dengan algoritma Harmony Search dalam jumlah titik yang banyak.

Oleh karena itu, dalam menggunakan program yang telah dibuat oleh penulis disarankan agar menginputkan nilai HMS dan NI yang cukup besar agar didapat rute yang lebih optimum.

DAFTAR RUJUKAN

Aldous, Joan M. and Wilson, Robin J. 2004. *Graphs and Applications An Introductory Approach.* Great Britain: Springer.

Purwanto. 1998. *Matematika Diskrit*. Malang: Institut Keguruan dan Ilmu Pendidikan Malang.

Satyananda, Darmawan. 2012. *Panduan Praktikum Struktur Data*. Buku tidak diterbitkan. Malang: Universitas Negeri Malang.

Maftuh, Ahmad Azami. 2010. *Study Penentuan Rute Busway yang Optimal Koridor Surabaya Timur-Barat dengan Metode Harmony Search*. Surabaya: Fakultas Teknologi Industri Institut Teknologi Sepuluh November Surabaya.

Wilson, R. J and Watkins, J.J. 1990. *Graph An Introductory Approach a First Course in Discrete Mathematics*. Canada: John Willy and Sons, Inc.

Yeun, Choong Liong and Zirour, Mourad. 2008. Vehicle Routing Problem: Models and Solution, *Journal Of Quality Measurement and Analysis*, 4(1):205-218, (Online)

Ralphsy, T.K., Kopmanz, L., Pulleyblankx, W.R. & Trotter, L.E. 2003. On the Capacitated Vehicle Routing Problem. *Mathematical Programming Series B 94343*. (Online), diakses tanggal 4 Maret 2015.

Hadwan, M., Ayob M., Sabar N.R., Qu, R. Tanpa Tahun. *A Harmony Search Algorithm for Nurse Rostering Problem*.(Online) diakses pada 02 Mei 2015.

Geem, Z.W., Kim J.H., Loganathan G.V. 2001. *A New Heuristic Optimization Algorithm : Harmony Search*. Journal of Simulation. hlm. 60-68.

Suthikamnarunai, N. 2008. *A Sweep Algorithm for Mix Fleet Vehicle Routing Problem*. Proceeding of The International MultiConference of Engineers and Computer Scientists, Vol. II. hlm. 19-21.