Penerapan algoritma *harmony search* (HS) pada *Multiple trip vehicle routing problem* (MTVRP) dan implementasinya

Nurul Istiyah[1], Sapti Wahyuningsih[2], Darmawan Satyananda[3]
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS NEGERI MALANG
Email: plimpy.nut@gmail.com

Abstrak: *Multiple Trip Vehicle Routing Problem* (MTVRP) adalah suatu permasalahan untuk mencari sejumlah rute minimum di mana setiap konsumen hanya dilayani tepat satu kali, proses pengiriman dimulai serta berakhir di depot yang sama dengan perluasan dan penambahan *multiple trip* pada setiap kendaraan ketika mendistribusikan barang serta *time window* pelayanan *customer*. Tahapan-tahapan algoritma *Harmony Search* (HS) diawali dengan tahap identifikasi masalah. Kemudian dilanjutkan ke tahap identifikasi parameter-parameter algoritma HS, inisialisasi *harmony memory* rute sementara, membangkitkan rute sementara baru secara random, meng-*update harmony memory* rute sementara, kemudian tahap terakhir yaitu mengecek kriteria pemberhentian. Untuk menyelesaikan permasalahan MTVRP dengan menggunakan algoritma HS, akan dibutuhkan banyak perulangan. Oleh sebab itu, untuk mempermudah pencarian rute, algoritma HS tersebut diimplementasikan ke dalam program komputer yang dibuat dengan Delphi 7.

Kata Kunci: *Multiple Trip Vehicle Routing Problem* (MTVRP), Algoritma *Harmony Search* (HS)

Abstract: *Multiple Trip Vehicle Routing Problem* (MTVRP) is a problem to find the minimum number of route that each customer only served exactly once and delivery process begins and ends at the same depot with the expansion and addition of *multiple trip* for each vehicle when distributing goods and *time window* customer service. The stages of the *Harmony Search* (HS) algorithm begins with the identification of problems. Then proceed to the next stage of identifying the parameters of HS algorithm, initializing of harmony memory of temporary route, generating a new temporary route randomly, updating harmony memory of temporary route, then the last stage is checking stop criteria. For completing the MTVRP problems by using HS algorithm, it will take a lot of repetition. Therefore, to simplify the search of route, the HS algorithm is implemented into a computer program using Delphi 7.

Keywords: *Multiple Trip Vehicle Routing Problem* (MTVRP), *Harmony Search* (HS) algorithm

Salah satu cabang matematika yang banyak membantu permasalahan dalam kehidupan sehari-hari adalah teori *graph*. Suatu permasalahan jika dimodelkan dalam teori *graph* akan dibuat sesederhana mungkin agar mudah dipahami, sehingga akan lebih mudah untuk diselesaikan. Salah satu permasalahan yang dapat diselesaikan dalam bentuk *graph* adalah masalah pengangkutan dan pengiriman barang dari produsen ke konsumen. Secara

1. Nurul Istiyah adalah mahasiswa jurusan Matematika FMIPA Universitas Negeri Malang
2. Sapti Wahyuningsih adalah dosen jurusan Matematika FMIPA Universitas Negeri Malang
3. Darmawan Satyananda adalah dosen jurusan Matematika FMIPA Universitas Negeri Malang

lebih khusus permasalahan pengangkutan dan pengiriman barang dapat dikategorikan sebagai permasalahan *Vehicle Routing Problem* (VRP). *Vehicle Routing Problem* (VRP) adalah permasalahan untuk mencari sejumlah rute minimum di mana setiap konsumen hanya dilayani tepat satu kali dan proses pengiriman dimulai serta berakhir di depot yang sama. *Multiple Trip Vehicle Routing Problem* (MTVRP) adalah pengembangan dari permasalahan VRP dengan perluasan dan penambahan *multiple trip* pada setiap kendaraan ketika mendistribusikan barang serta *time window* pelayanan *customer*.

Beberapa algoritma dan metode dari MTVRP yang telah dibahas sebelumnya yaitu *Metode Insertion Heuristic* (Oktaviani, 2010) yang memiliki kelebihan yaitu pemilihan dan penyisipan *customer* pada rute yang didasarkan pada minimum waktu tempuh dan *Algoritma Sequential Insertion* (Yunita, 2013) yang memiliki kelebihan yaitu pemilihan *customer* pertama (*seed customer*) didasarkan pada jarak terjauh dari depot serta pemilihan *customer* selanjutnya didasarkan pada jarak terdekat dari *seed customer*.

Aulia (2012) mengungkapkan bahwa algoritma *Harmony Search* (HS) adalah salah satu algoritma metaheuristik yang diusulkan oleh Zong Woo Geem dalam jurnal berjudul *"A New Heuristic Optimization Algorithm: Harmony Search"* tahun 2001. Algoritma tersebut terinspirasi oleh proses pertunjukan musik. Dalam proses tersebut, dianalogikan seorang musisi mengimprovisasi *pitch instrument* di mana proses tersebut bertujuan untuk mendapatkan keadaan terbaik berdasarkan perkiraan estetika. Harmoni dalam musik tersebut merupakan representasi dari vektor solusi sedangkan proses improvisasinya merepresentasikan pencarian global atau lokal dalam teknik optimasi. Algoritma *Harmony Search* memiliki struktur yang relatif mudah karena tidak perlu melibatkan kalkulasi matematika yang kompleks.

Hadwan (2013) menuliskan langkah-langkah dari algoritma HS secara umum adalah sebagai berikut:

Langkah 1 : Inisialisasi masalah dan parameter algoritma HS.
Langkah 2 : Inisialisasi *harmony memory* (HM) rute sementara.
Langkah 3 : Membangkitkan rute sementara yang baru.
Langkah 4 : Meng-update *harmony memory* (HM) rute sementara.
Langkah 5 : Ulangi langkah 3 dan 4 hingga kriteria pemberhentian tercapai, yaitu jika iterasi yang dilakukan sama dengan jumlah iterasi yang diinputkan.

Berdasarkan uraian di atas, akan dibahas mengenai algoritma Harmony Search (HS) pada permasalahan Multiple Trip Vehicle Routing Problem (MTVRP).

HASIL YANG DIHARAPKAN

1. Mengetahui langkah-langkah algoritma *Harmony Search* (HS) dalam menyelesaikan MTVRP.
2. Menerapkan algoritma Harmony Search (HS) pada contoh permasalahan MTVRP.
3. Mengimplementasikan algoritma *Harmony Search* (HS) ke dalam program komputer yang dibuat dengan Delphi 7.

HASIL DAN PEMBAHASAN

- **Multiple Trip Vehicle Routing Problem (MTVRP)**

  Multiple Trip Vehicle Routing Problem (MTVRP) adalah pengembangan dari permasalahan Vehicle Routing Problem (VRP) dengan perluasan dan penambahan *multiple trip* pada setiap kendaraan ketika mendistribusikan barang serta *time window* pelayanan *customer*. Tujuan dari permasalahan ini adalah meminimumkan jumlah

kendaraan dengan mengoptimalkan sejumlah rute yang memenuhi kendala. Sehingga satu kendaraan dapat menempuh lebih dari satu rute sekaligus.

Formulasi matematika MTVRP (Bahar, 2003) adalah sebagai berikut:

a) Fungsi Tujuan: Untuk meminimasi total jarak tempuh.
   $min \sum_{k \in K} \sum_{r \in R^k} c_r^k \, x_r^k$
b) Untuk menjamin bahwa seluruh *customer* dapat dilayani.
   $\sum_{k \in K} \sum_{r \in R^k} A_{ri}^k \, x_r^k \geq 1 \quad \forall i \in N$
c) Untuk memastikan bahwa total waktu pendistribusian suatu kendaraan tidak melebihi waktu maksimum yang diberikan.
   $\sum_{r \in R^k} T_r^k \, x_r^k \leq T_{MAX} \quad \forall k \in K$
d) Untuk menjamin bahwa setiap kendaraan melewati 1 atau lebih dari 1 rute:
   $\sum x_r^k \geq 1 \quad \forall k \in K$
e) Untuk menjamin variabel keputusan x_r^k merupakan integer biner.
   $x_r^k \in \{0, 1\} \quad \forall k \in K, \forall r \in R^k$

Keterangan formula:

K : himpunan kendaraan pada depot, dengan $K = \{1, 2, \dots, k\}$
R^k : himpunan rute yang ditempuh kendaraan k, dengan $R = \{R^1, R^2, \dots, R^k\}$
V : himpunan titik
q_j : jumlah permintaan *customer* j
r : indeks rute
c_r^k : jarak tempuh kendaraan k ketika menempuh rute r
A_{ri}^k : konstanta, bernilai 1 jika kendaraan k dengan rute r menuju *customer* i, dan bernilai 0 untuk kondisi lainnya
T_r^k : waktu tempuh kendaraan untuk menempuh rute r
T_{MAX} : waktu tempuh maksimum untuk suatu kendaraan
x_r^k : integer biner, bernilai 1 jika kendaraan k menggunakan rute r, dan bernilai 0 untuk kondisi lainnya

- **Algoritma Harmony Search (HS) pada Multiple Trip Vehicle Routing Problem (MTVRP)**

Terdapat 6 tahapan dalam algoritma HS, yaitu identifikasi masalah, identifikasi parameter algoritma HS, inisialisasi *harmony memory* rute sementara, membangkitkan rute sementara baru, meng-*update harmony memory* rute sementara, dan mengecek kriteria pemberhentian.

1. Identifikasi Masalah
   Meminimalkan total jarak tempuh $f(x_i)$
   dengan $x_i \in X_i \, , i = 1, 2, \dots, N; \; BB \leq X_i \leq BA$
   di mana $f(x_i)$ 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. Identifikasi Parameter Algoritma HS

Agar *Harmony Memory* (HM) rute sementara dapat digunakan secara efektif, algoritma HS mengadopsi beberapa parameter sebagai berikut:

a. *Harmony Memory Size* (HMS)
HMS adalah banyaknya rute sementara yang terbentuk.

b. *Harmony Memory Consideration Rate* ($HMCR$)
$HMCR$ bernilai $0 \leq HMCR \leq 1$. Umumnya berkisar antara 0,7 sampai 0,95 (Maftuh, 2010).

c. Bandwidth (bw)
Bandwidth (bw) pada MTVRP adalah sebarang bilangan bulat positif. *Bandwidth* berguna untuk menyesuaikan *customer*. Persamaan (1) berikut ini adalah rumus penyesuaian *customer* :

$$NP = x_{old} + bw \times \varepsilon \quad (1)$$

di mana NP adalah nilai penyesuaian *customer* baru
x_{old} adalah *customer* yang tersimpan pada *harmony 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
Kriteria pemberhentian tercapai jika iterasi yang dilakukan sama dengan jumlah iterasi yang diinputkan.

3. Inisialisasi *Harmony Memory* (HM) Rute Sementara

Persamaan (2) di bawah ini merepresentasikan inisialisasi HM yang didapatkan dengan membangkitkan matriks *harmony memory* rute sementara yang berisi rute sementara sebanyak HMS.

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

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.

4. Membangkitkan Rute Sementara Baru

Proses pembangkitan rute sementara baru, dilakukan dengan merandom bilangan bulat dengan selang 1 hingga banyaknya *customer*, kemudian menyesuaikannya dengan nilai $HMCR$ dan PAR, seperti berikut:

- Apabila bilangan *random* r_1 yang dibangkitkan lebih besar dari $HMCR$ maka akan dipilih *customer* yang baru secara *random*.
  $CB(i) = rand[BB, BA]$
  di mana $CB(i) =$ *customer* baru ke i
  $BB =$ batas bawah *customer*
  $BA =$ batas atas *customer*

- Apabila bilangan *random* r_1 yang dibangkitkan lebih kecil dari $HMCR$ dan pembangkitan bilangan *random* r_2 berikutnya lebih besar dari PAR, maka *customer* ke-i akan diambil secara *random* dari HM dengan $CB(i) \in \{x_i^1, x_i^2, \dots, x_i^{HMS-1}, x_i^{HMS}\}$ di mana $CB(i) =$ *customer* baru ke i
- Apabila bilangan *random* r_1 yang dibangkitkan lebih kecil dari $HMCR$ dan pembangkitan bilangan *random* r_2 berikutnya lebih kecil dari PAR, maka terdapat penyesuaian terhadap *customer* ke-i yang akan diambil dari HM, yaitu:
  1. Lakukan random *customer* dengan $CBS(i) \in \{x_i^1, x_i^2, \dots, x_i^{HMS-1}, x_i^{HMS}\}$ di mana $CBS(i) =$ *customer* baru sementara ke i
  2. Hitung nilai penyesuaian *customer* baru dengan rumus sebagai berikut:
     $NP = CBS(i) + (bw \times \varepsilon)$
     di mana $NP =$ nilai penyesuaian *customer* baru
     $bw =$ sebarang bilangan bulat positif
     $\varepsilon =$ 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 $CB(i) = CBS(i)$.

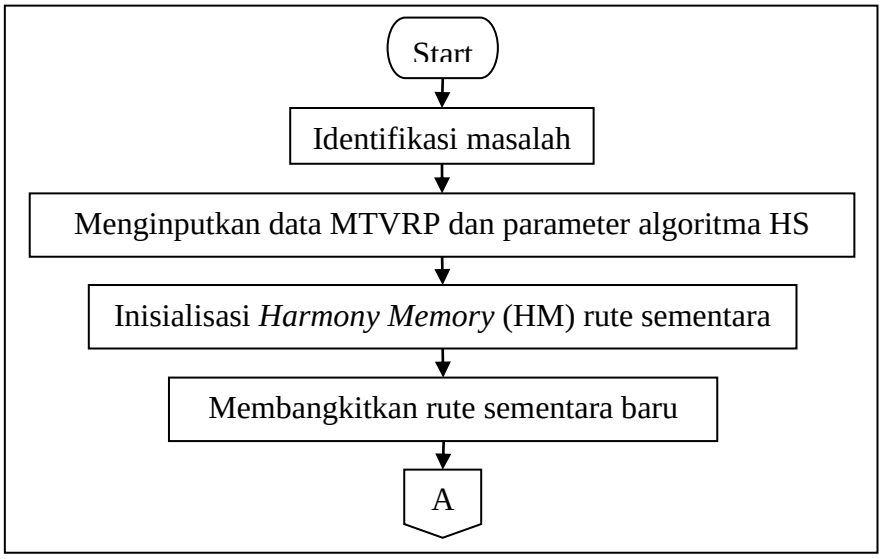
5. Meng-*update Harmony Memory* (HM) Rute Sementara

   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.

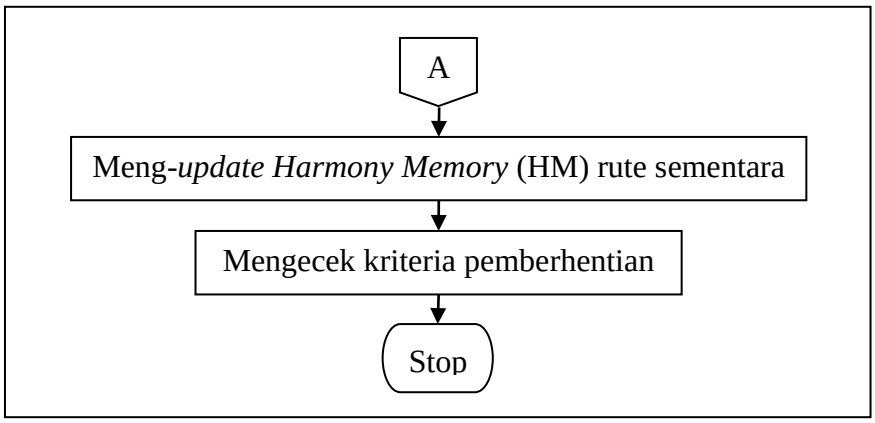
6. Mengecek Kriteria Pemberhentian (*Check Stopping Criterion*)

   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 adalah *Block* Diagram dari algoritma *Harmony Search* pada MTVRP.



Gambar 1 : *Block* Diagram (Bagian 1)



Gambar 2 : *Block* Diagram (Bagian 2)

Kelebihan dari algoritma ini terletak pada proses pembentukan rute yang dilakukan secara random.

- **Contoh Penerapan**

Berikut ini merupakan permasalahan distribusi yang dirujuk dari skripsi tahun 2013 oleh saudari Nine Winda Yunita dengan hasil rute $0-7-4-6-0$, $0-1-8-2-3-0$, dan $0-5-0$ dengan $T_{\text{total}} = 1{,}6172$ jam dan total jarak tempuh 95 km.

Permasalahan pada Contoh 1 di atas adalah sebagai berikut: Suatu perusahaan A akan melakukan pemuatan dan pembongkaran barang ke *outlet-outlet* yang tersebar di berbagai daerah. Pelayanan dilakukan selama 1 jam dengan kapasitas tiap kendaraan sebesar 65 kardus, kecepatan rata-rata 60 km/jam, dan waktu pelayanan tiap tabungnya selama 0,87 detik/kardus atau 0,0002417 jam/kardus. Adapun daftar permintaan tiap-tiap *outlet* diberikan pada Tabel 3.1 berikut.

Tabel 1 : Jumlah permintaan masing-masing *outlet*

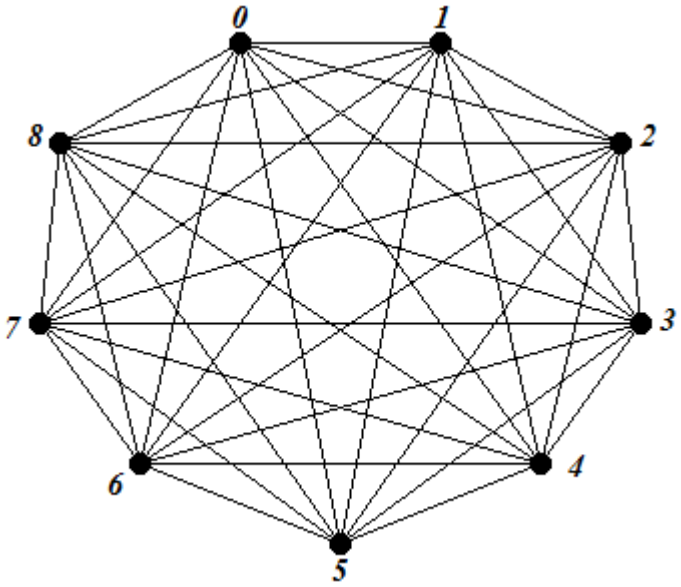
Outlet	1	2	3	4	5	6	7	8
Permintaan	15	20	10	25	15	10	25	20

Tabel 2 : Jarak antara depot ke *outlet* dan antar *outlet*

c_{ij}	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
0	0	5	8	11	16	13	12	9	14
1	5	0	5	7	12	11	15	13	8
2	8	5	0	6	5	3	6	4	11
3	11	7	6	0	11	3	7	14	15
4	16	12	5	11	0	8	6	1	17
5	13	11	3	3	8	0	9	11	10
6	12	15	6	7	6	9	0	14	6
7	9	13	4	14	1	11	14	0	10
8	14	8	11	15	17	10	6	10	0

Keterangan:

Untuk kolom pertama dan baris pertama, 0 menyatakan depot dan 1,2,… 8 merupakan outlet. Jarak dari 0 (depot) ke outlet 1 adalah 5 km, jarak dari depot ke outlet 2 adalah 8 km, dan seterusnya.



Gambar 3 : Model *Graph* K_9

Pada Gambar 3, lokasi depot dan *outlet-outlet* diwakili oleh titik dan jalan yang dilalui dari depot menuju *outlet* atau dari suatu *outlet* ke *outlet* lainnya diwakili oleh sisi.

Untuk menghitung waktu pelayanan dari depot ke *outlet* dan antar *outlet*, digunakan rumus (3) berikut:

$$t_{ij} = \left(c_{ij} \div V_k\right) + \left(T_s \times q_j\right) \quad (3)$$

di mana t_{ij} adalah waktu pelayanan dari titik i ke titik j
c_{ij} adalah jarak dari titik i ke titik j
V_k adalah kecepatan rata-rata kendaraan
T_s adalah waktu pelayanan per kardus
q_j adalah permintaan di titik j

Tabel 3 : Waktu pelayanan antara depot ke *outlet* dan antar *outlet*

t_{ij}	0	1	2	3	4	5	6	7	8
0	0	0,09	0,14	0,19	0,27	0,22	0,20	0,16	0,24
1	0,08	0	0,09	0,12	0,21	0,19	0,25	0,22	0,14
2	0,13	0,09	0	0,10	0,09	0,05	0,10	0,07	0,19
3	0,18	0,12	0,10	0	0,19	0,05	0,12	0,24	0,25
4	0,27	0,20	0,09	0,19	0	0,14	0,10	0,02	0,29
5	0,22	0,19	0,05	0,05	0,14	0	0,15	0,19	0,17
6	0,20	0,25	0,10	0,12	0,11	0,15	0	0,24	0,10
7	0,15	0,22	0,07	0,24	0,02	0,19	0,24	0	0,17
8	0,23	0,14	0,19	0,25	0,29	0,17	0,10	0,17	0

Keterangan:
Untuk kolom pertama dan baris pertama, 0 menyatakan depot dan 1,2,… 8 merupakan outlet. Waktu pelayanan dari 0 (depot) ke outlet 1 adalah 0,09 jam, jarak dari depot ke outlet 2 adalah 0,14 jam, dan seterusnya.

Berikut ini adalah penyelesaian permasalahan distribusi tersebut dengan menggunakan algoritma *Harmony Search* (HS).

1. Identifikasi Masalah

   Permasalahan yang akan di selesaikan adalah MTVRP dengan fungsi tujuan meminimasi total jarak tempuh. Sedangkan kendala yang muncul adalah sebagai berikut:

   - Kecepatan rata-rata kendaraan $V_k = 60$ km/jam
   - Jumlah permintaan tiap *outlet*
   - Kapasitas maksimal kendaraan $Q = 65$ kardus
   - Waktu pelayanan per kardus $T_s = 0{,}87$ detik/kardus
     $= 0{,}0002417$ jam/kardus
   - Batas waktu $T_w = 1$ jam

2. Identifikasi Parameter Algoritma HS

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

   - Nilai $HMCR$ dipilih antara $[0.7, 0.95]$, terpilih $HMCR = 0{,}87$
   - Nilai PAR dipilih antara $[0.1, 0.5]$, terpilih $PAR = 0{,}33$
   - Nilai $HMS = 3$
   - $bw = 10$
   - Kriteria pemberhentian dipilih 1 iterasi, $NI = 1$

3. Inisialisasi *Harmony Memory* (HM) Rute Sementara

   Harmony Memory (HM) rute sementara dibangun sebanyak HMS yang sudah ditentukan. HM merupakan matriks yang berisi sekumpulan rute sementara yang dibangun secara acak. Berikut merupakan HM rute sementara yang dibangkitkan sebanyak HMS:

$$HM = \left[\begin{array}{cccccccc|c} 2 & 4 & 5 & 1 & 3 & 7 & 6 & 8 & f(x_i) = 114 \\ 1 & 4 & 7 & 2 & 3 & 5 & 8 & 6 & f(x_i) = 92 \\ 4 & 3 & 1 & 5 & 8 & 2 & 6 & 7 & f(x_i) = 119 \end{array}\right]$$

   Dengan $X_1 = [2 \quad 4 \quad 5 \quad 1 \quad 3 \quad 7 \quad 6 \quad 8]$
   $X_2 = [1 \quad 4 \quad 7 \quad 2 \quad 3 \quad 5 \quad 8 \quad 6]$
   $X_3 = [4 \quad 3 \quad 1 \quad 5 \quad 8 \quad 2 \quad 6 \quad 7]$

   Berikut ini adalah proses pembentukan rute untuk menghitung total jarak tempuh masing-masing rute sementara:

   1) Total jarak tempuh untuk $X_1 = [2 \quad 4 \quad 5 \quad 1 \quad 3 \quad 7 \quad 6 \quad 8]$
      - Langkah 1: Bentuk rute yang berawal dan berakhir di depot
        $Rute(1) = [0, 0]$
      - Langkah 2: Pembentukan rute
        - Sisipkan *outlet* 2 ke dalam rute, sehingga
          $Rute(1) = [0, 2, 0]$
          Update rute sementara $X_1 = [4 \quad 5 \quad 1 \quad 3 \quad 7 \quad 6 \quad 8]$
          Kendala kapasitas: $q_2 = 20 < Q$ (memenuhi)
          Kendala waktu: $t_{02} + t_{20} = 0{,}27 < T_w$ (memenuhi)
          Karena kedua kendala memenuhi, lanjutkan ke penyisipan berikutnya.
        - Sisipkan *outlet* 4 ke dalam rute, sehingga
          $Rute(1) = [0, 2, 4, 0]$

Update rute sementara $X_1 = [5 \quad 1 \quad 3 \quad 7 \quad 6 \quad 8]$
Kendala kapasitas: $q_2 + q_4 = 45 < Q$ (memenuhi)
Kendala waktu: $t_{02} + t_{24} + t_{40} = 0{,}49 < T_w$ (memenuhi)
Karena kedua kendala memenuhi, lanjutkan ke penyisipan berikutnya.

- Sisipkan *outlet* 5 ke dalam rute, sehingga
  $Rute(1) = [0, 2, 4, 5, 0]$
  Update rute sementara $X_1 = [1 \quad 3 \quad 7 \quad 6 \quad 8]$
  Kendala kapasitas: $q_2 + q_4 + q_5 = 60 < Q$ (memenuhi)
  Kendala waktu: $t_{02} + t_{24} + t_{45} + t_{50} = 0{,}58 < T_w$ (memenuhi)
  Karena kedua kendala memenuhi, lanjutkan ke penyisipan berikutnya.
- Sisipkan *outlet* 1 ke dalam rute, sehingga
  $Rute(1) = [0, 2, 4, 5, 1, 0]$
  Update rute sementara $X_1 = [3 \quad 7 \quad 6 \quad 8]$
  Kendala kapasitas: $q_2 + q_4 + q_5 + q_1 = 75 > Q$ (tidak memenuhi)
  Kendala waktu: $t_{02} + t_{24} + t_{45} + t_{51} + t_{10} = 0{,}63 < T_w$ (memenuhi)
  Karena kendala kapasitas tidak memenuhi, penyisipan *outlet* 1 dibatalkan.
  Update rute sementara $X_1 = [1 \quad 3 \quad 7 \quad 6 \quad 8]$
  Diperoleh $Rute(1) = [0, 2, 4, 5, 0]$.

Lanjutkan untuk pembentukan rute berikutnya dengan rute sementara $X_1 = [1 \quad 3 \quad 7 \quad 6 \quad 8]$.

- Langkah 1: Bentuk rute yang berawal dan berakhir di depot
  $Rute(2) = [0, 0]$
- Langkah 2: Pembentukan rute
  - Sisipkan *outlet* 1 ke dalam rute, sehingga
    $Rute(2) = [0, 1, 0]$
    Update rute sementara $X_1 = [3 \quad 7 \quad 6 \quad 8]$
    Kendala kapasitas: $q_1 = 15 < Q$ (memenuhi)
    Kendala waktu: $t_{01} + t_{10} = 0{,}17 < T_w$ (memenuhi)
    Karena kedua kendala memenuhi, lanjutkan ke penyisipan berikutnya.
  - Sisipkan *outlet* 3 ke dalam rute, sehingga
    $Rute(2) = [0, 1, 3, 0]$
    Update rute sementara $X_1 = [7 \quad 6 \quad 8]$
    Kendala kapasitas: $q_1 + q_3 = 25 < Q$ (memenuhi)
    Kendala waktu: $t_{01} + t_{13} + t_{30} = 0{,}39 < T_w$ (memenuhi)
    Karena kedua kendala memenuhi, lanjutkan ke penyisipan berikutnya.
  - Sisipkan *outlet* 7 ke dalam rute, sehingga
    $Rute(2) = [0, 1, 3, 7, 0]$
    Update rute sementara $X_1 = [6 \quad 8]$
    Kendala kapasitas: $q_1 + q_3 + q_7 = 50 < Q$ (memenuhi)
    Kendala waktu: $t_{01} + t_{13} + t_{37} + t_{70} = 0{,}6 < T_w$ (memenuhi)
    Karena kedua kendala memenuhi, lanjutkan ke penyisipan berikutnya.
  - Sisipkan *outlet* 6 ke dalam rute, sehingga
    $Rute(2) = [0, 1, 3, 7, 6, 0]$
    Update rute sementara $X_1 = [8]$
    Kendala kapasitas: $q_1 + q_3 + q_7 + q_6 = 60 < Q$ (memenuhi)
    Kendala waktu: $t_{01} + t_{13} + t_{37} + t_{76} + t_{60} = 0{,}88 < T_w$ (memenuhi)
    Karena kedua kendala memenuhi, lanjutkan ke penyisipan berikutnya.

- Sisipkan *outlet* 8 ke dalam rute, sehingga
  $Rute(2) = [0, 1, 3, 7, 6, 8, 0]$
  Update rute sementara $X_1 = [\ ]$
  Kendala kapasitas: $q_1 + q_3 + q_7 + q_6 + q_8 = 80 > Q$
  (tidak memenuhi)
  Kendala waktu: $t_{01} + t_{13} + t_{37} + t_{76} + t_{68} + t_{80} = 1{,}02 > T_w$
  (tidak memenuhi)
  Karena kedua kendala tidak memenuhi, penyisipan *outlet* 8 dibatalkan.
  Update rute sementara $X_1 = [8]$
  Diperoleh $Rute(2) = [0, 1, 3, 7, 6, 0]$.

Lanjutkan untuk pembentukan rute berikutnya dengan rute sementara $X_1 = [8]$.

- Langkah 1: Bentuk rute yang berawal dan berakhir di depot
  $Rute(3) = [0, 0]$
- Langkah 2: Pembentukan rute
  - Sisipkan *outlet* 8 ke dalam rute, sehingga
    $Rute(3) = [0, 8, 0]$
    Update rute sementara $X_1 = [\ ]$
    Kendala kapasitas: $q_8 = 20 < Q$ (memenuhi)
    Kendala waktu: $t_{08} + t_{80} = 0{,}47 < T_w$ (memenuhi)
    Karena kedua kendala memenuhi dan semua *outlet* telah disisipkan maka proses pembentukan rute selesai.
    Diperoleh $Rute(3) = [0, 8, 0]$.

Sehingga pada rute sementara pertama diperoleh:
$Rute(1) = [0, 2, 4, 5, 0]$, dengan jarak tempuh 34 km.
$Rute(2) = [0, 1, 3, 7, 6, 0]$, dengan jarak tempuh 52 km.
$Rute(3) = [0, 8, 0]$, dengan jarak tempuh 28 km.
Jadi, total jarak tempuh $f(x_i) = 114\ km$.

2) Total jarak tempuh untuk $X_2 = [1 \quad 4 \quad 7 \quad 2 \quad 3 \quad 5 \quad 8 \quad 6]$
Dengan proses yang sama, diperoleh:
$Rute(1) = [0, 1, 4, 7, 0]$, dengan jarak tempuh 27 km.
$Rute(2) = [0, 2, 3, 5, 8, 0]$, dengan jarak tempuh 41 km.
$Rute(3) = [0, 6, 0]$, dengan jarak tempuh 24 km.
Jadi, total jarak tempuh $f(x_i) = 92\ km$.

3) Total jarak tempuh untuk $X_3 = [4 \quad 3 \quad 1 \quad 5 \quad 8 \quad 2 \quad 6 \quad 7]$
Dengan proses yang sama, diperoleh:
$Rute(1) = [0, 4, 3, 1, 5, 0]$, dengan jarak tempuh 58 km.
$Rute(2) = [0, 8, 2, 6, 0]$, dengan jarak tempuh 43 km.
$Rute(3) = [0, 7, 0]$, dengan jarak tempuh 18 km.
Jadi, total jarak tempuh $f(x_i) = 119\ km$.

4. Membangkitkan Rute Sementara yang Baru

Pada langkah 4 ini, akan dibangkitkan rute sementara yang baru yaitu seperti berikut:
$X' = [x'_1 \quad x'_2 \quad x'_3 \quad x'_4 \quad x'_5 \quad x'_6 \quad x'_7 \quad x'_8]$.

1) Menentukan nilai x_1'
   Dibangkitkan bilangan *random* misalnya $r_1 = 0{,}9153$.
   Karena $r_1 > HMCR$, maka *outlet* dipilih secara acak dari $\{1, 2, 3, 4, 5, 6, 7, 8\}$ atau $x_1' \in \{1, 2, 3, 4, 5, 6, 7, 8\}$.
   $$x_1' = 3$$
   Jadi, $x_1' = 3$.
   $X' = [3]$
   Outlet yang belum terpilih $= \{1, 2, 4, 5, 6, 7, 8\}$

2) Menentukan nilai x_2'
   Dibangkitkan bilangan *random* misalnya $r_1 = 0{,}7237$.
   Karena $r_1 < HMCR$, maka *outlet* dipilih secara acak dengan $x_2' \in \{x_2^1, x_2^2, x_2^3\} = \{3, 4\}$ dan dengan syarat $x_2' \neq x_1'$.
   $$x_2' = 4$$
   Selanjutnya, dibangkitkan bilangan *random* misalnya $r_2 = 0{,}4957$.
   Karena $r_2 > PAR$, maka *outlet* $x_2' = 4$ tetap dipertahankan.
   Jadi, $x_2' = 4$.
   $X' = [3 \quad 4]$
   Outlet yang belum terpilih $= \{1, 2, 5, 6, 7, 8\}$

3) Menentukan nilai x_3'
   Dibangkitkan bilangan *random* misalnya $r_1 = 0{,}4083$.
   Karena $r_1 < HMCR$, maka *outlet* dipilih secara acak dengan $x_3' \in \{x_3^1, x_3^2, x_3^3\} = \{1, 5, 7\}$ dan dengan syarat $x_3' \neq x_1', x_2'$.
   $$x_3' = 1$$
   Selanjutnya, dibangkitkan bilangan *random* misalnya $r_2 = 0{,}1803$.
   Karena $r_2 < PAR$, maka akan dilakukan penyesuaian *outlet* sebagai berikut.
   $\varepsilon = random[-1, 1] = 0{,}8224$
   $D = 1 + bw \times \varepsilon = 1 + 10 \times 0{,}8224 = 9{,}224$
   Karena $D > 8 = BA$, maka $x_3' = BA = 8$.
   Jadi, $x_3' = 8$.
   $X' = [3 \quad 4 \quad 8]$
   Outlet yang belum terpilih $= \{1, 2, 5, 6, 7\}$

Ulangi langkah-langkah tersebut hingga semua *outlet* termuat dalam rute.
Diperoleh rute sementara yang baru $X' = [3 \quad 4 \quad 8 \quad 1 \quad 2 \quad 5 \quad 6 \quad 7]$

Selanjutnya, akan dihitung total jarak tempuh dari rute sementara yang baru dengan terlebih dahulu dilakukan proses pembentukan rute seperti proses pada HM rute sementara, sehingga diperoleh:
$Rute(1) = [0, 3, 4, 8, 0]$, dengan jarak tempuh 53 km.
$Rute(2) = [0, 1, 2, 5, 6, 0]$, dengan jarak tempuh 34 km.
$Rute(3) = [0, 7, 0]$, dengan jarak tempuh 18 km.
Jadi, total jarak tempuh $f(x') = 105\ km$.

5. Meng-*update Harmony Memory* (HM) Rute Sementara

Karena total jarak tempuh rute sementara yang baru lebih pendek dari rute sementara dengan total jarak tempuh terpanjang pada HM, maka rute sementara yang

baru dimasukkan ke dalam HM menggantikan rute sementara dengan total jarak tempuh terpanjang.

Sehingga *Harmony Memory* (HM) rute sementara menjadi:

$$HM = \begin{bmatrix} 2 & 4 & 5 & 1 & 3 & 7 & 6 & 8 \\ 1 & 4 & 7 & 2 & 3 & 5 & 8 & 6 \\ 3 & 4 & 8 & 1 & 2 & 5 & 6 & 7 \end{bmatrix} \begin{matrix} f(x_1) = 114 \\ f(x_2) = 92 \\ f(x_3) = 105 \end{matrix}$$

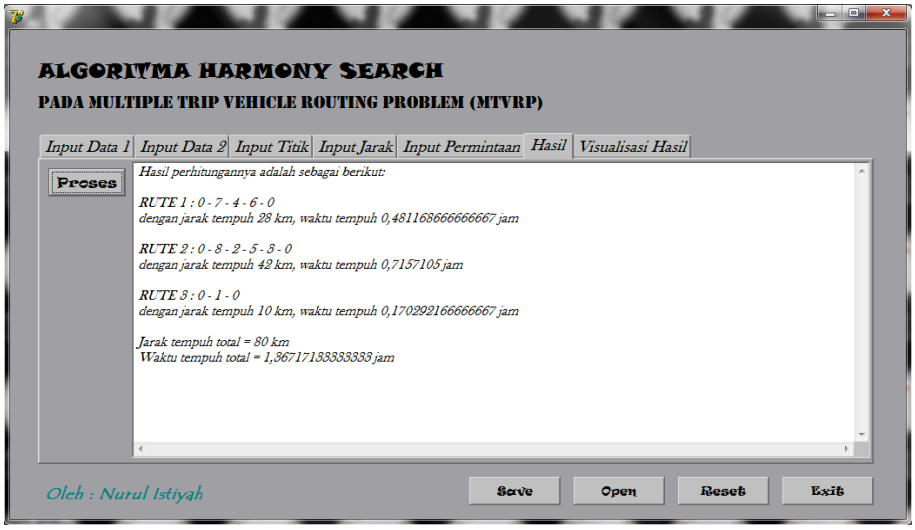
6. Mengecek Kriteria Pemberhentian (*Check Stopping Criterion*)

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

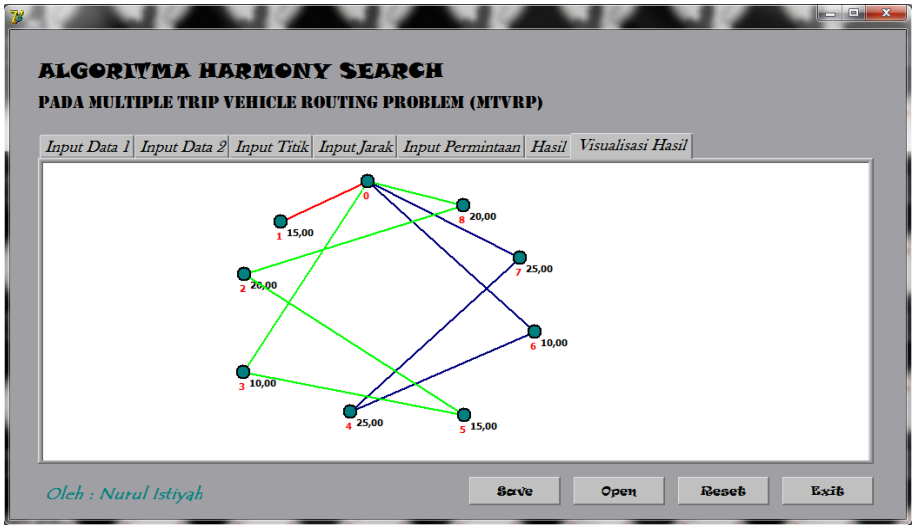
Berdasarkan hasil penyelesaiannya, diperoleh rute pertama $0 - 1 - 4 - 7 - 0$, rute kedua $0 - 2 - 3 - 5 - 8 - 0$, dan rute ketiga $0 - 6 - 0$ dengan $T_{total} = 1{,}57$ jam dan total jarak tempuh adalah 92 km. Karena $T_{total} = 1{,}57 > T_w$ sehingga dibutuhkan lebih dari satu kendaraan yaitu 2 kendaraan.

Algoritma HS diimplementasikan ke dalam program komputer. Untuk menjalankannya, diawali dengan menginputkan data seperti banyaknya titik dengan titik 0 sebagai depot dan titik lainnya sebagai *customer*, jarak antar titik, permintaan tiap *customer*, kapasitas kendaraan, kecepatan rata-rata kendaraan, waktu pelayanan tiap item, *time window* dan parameter-parameter algoritma HS. Data tersebut di atas akan diproses sesuai dengan tahapan algoritma HS yaitu identifikasi masalah, identifikasi parameter-parameter algoritma HS, inisialisasi *harmony memory* rute sementara, membangkitkan rute sementara baru, meng-*update harmony memory* rute sementara dan mengecek kriteria pemberhentian. Output dari proses tersebut yaitu urutan rute, total jarak tempuh dan total waktu tempuh dari masing-masing rute berupa teks tulisan serta urutan rute berupa visualisasi gambar.

Berikut ini adalah penyelesaian permasalahan distribusi tersebut dengan menggunakan program.



Gambar 4 : Tampilan hasil perhitungan program



Gambar 5 : Tampilan visualisasi hasil

Berdasarkan Gambar 4 di atas, diperoleh hasil berupa urutan rute, yaitu sebagai berikut:

1. Rute 1 : 0 – 7 – 4 – 6 – 0, dengan jarak tempuh 28 km dan waktu tempuh 0,481168666666667 jam,
2. Rute 2 : 0 – 8 – 2 – 5 – 3 – 0, dengan jarak tempuh 42 km dan waktu tempuh 0,7157105 jam, dan
3. Rute 3 : 0 – 1 – 0, dengan jarak tempuh 10 km dan waktu tempuh 0,170292166666667 jam.

Sehingga diperoleh total jarak tempuh adalah 80 km dan total waktu tempuh adalah 1,36717133333333 jam. Sedangkan Gambar 5 adalah hasil gambar rute yang diperoleh pada hasil perhitungan Gambar 4.

Contoh permasalahan MTVRP di atas juga diselesaikan dengan menggunakan program *sequential insertion* yang dirancang oleh Yunita (2013) menghasilkan 3 rute dengan total jarak tempuh 95 km dan waktu tempuh total 1,612 jam. Sehingga contoh permasalahan tersebut menunjukkan bahwa program HS yang telah dibuat dapat menghasilkan rute dengan total jarak tempuh yang lebih pendek dibandingkan rute yang dihasilkan program *sequential insertion*.

Program *harmony search* (HS) ini telah diuji coba dengan menggunakan 9, 15, 20, 40, 80, dan 160 titik.

KESIMPULAN

Berdasarkan hasil dan pembahasan, berikut ini adalah beberapa kesimpulan yang bisa diberikan:

1. Tahapan-tahapan algoritma *Harmony Search* (HS) diawali dengan tahap identifikasi masalah. Tahap selanjutnya yaitu mengidentifikasi parameter-parameter algoritma HS yaitu NI, HMS, bw, $HMCR$ dan PAR. Setelah itu masuk ke tahap selanjutnya yaitu inisialisasi *harmony memory* rute sementara, membangkitkan rute sementara baru, meng-*update harmony memory* rute sementara dan tahap terakhir yaitu mengecek kriteria pemberhentian.
2. Dengan menerapkan algoritma HS pada permasalahan MTVRP, diperoleh 3 rute dengan $T_{total} = 1{,}57$ jam dan total jarak tempuh adalah 92 km. Karena $T_{total} = 1{,}57 \geq T_w$ sehingga dibutuhkan 2 kendaraan.

3. Untuk menjalankan program yang telah dibuat, diawali dengan menginputkan data seperti banyaknya titik, jarak antar titik, permintaan tiap *customer*, kapasitas kendaraan, kecepatan rata-rata kendaraan, waktu pelayanan tiap item, *time window* dan parameter-parameter algoritma HS. Data tersebut diproses sesuai dengan tahapan algoritma HS dan menghasilkan output yaitu urutan rute, total jarak tempuh dan total waktu tempuh dari masing-masing rute berupa teks tulisan serta urutan rute berupa visualisasi gambar.

SARAN

Program yang telah dibuat ini dapat digunakan untuk menyelesaikan permasalahan MTVRP pada kegiatan Praktik Kerja Lapangan (PKL), karena telah diuji hingga 160 titik. Agar diperoleh rute dengan total jarak tempuh yang minimal, disarankan agar jumlah iterasi yang diinputkan (NI) lebih dari atau sama dengan faktorial dari jumlah *customer*.

Pada penelitian selanjutnya disarankan untuk menambahkan teknik-teknik yang dapat memberikan kestabilan hasil dalam setiap iterasi dan memperkuat nilai solusinya dengan mempertimbangkan tingkat kemampuan komputer atau dengan menambahkan procedure untuk menghitung kecepatan komputasi program dari masing-masing algoritma, sehingga dapat dibandingkan program dengan algoritma manakah yang memiliki waktu komputasi lebih cepat.

DAFTAR PUSTAKA

Aulia, I., Nababan, E. B. & Muchtar, M. A. 2012. *Penerapan Harmony Search Algorithm dalam Permasalahan Penjadwalan Flow Show*. Jurnal Dunia Teknologi Informasi, Vol. 1, No. 1: hlm.1 – 7, (Online), (http://jurnal.usu.ac.id/index.php/duniait/article/view/407/210), diakses 25 Maret 2014.

Bahar, Emirul. 2003. *Analisis Penentuan Jalur Transportasi Limbah Minyak pada Aktivitas Pelayaran Laut untuk Menghasilkan Total Biaya Pelayaran Minimum*. Jurnal Ilmiah Ekonomi & Bisnis (Jurnal Ilmiah Ekonomi Bisnis), Vol. 8, No. 2: hlm. 88.

Hadwan, M., Ayob, M., Sabar N. R. & Qu, Roug. 2013. *A Harmony Search Algorithm for Nurse Rostering Problems*. Information Science. (Online) (http://www.cs.nott.ac.uk/~rxq/files/INS13hsa.pdf), diakses 8 April 2014.

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 Nopember Surabaya.

Oktaviani, Gladis Dwi. 2010. *Implementasi Metode Insertion Heuristic dalam Penyelesaian Multiple Trip Vehicle Routing Problem (MTVRP) dan Analisanya*. Skripsi tidak diterbitkan. Malang: Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Malang.

Yunita, Nine Winda. 2013. *Algoritma Sequential Insertion untuk Menyelesaikan Masalah Multiple Trip Vehicle Routing Problem (MTVRP)*. Skripsi tidak diterbitkan. Malang: Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Negeri Malang.