BAB III
MODEL OPTIMISASI VRPTW DAN PENYELESAIANNYA MENGGUNAKAN ALGORITMA HBMO

3.1 Deskripsi Masalah

Penelitian ini membahas tentang *Vehicle Routing Problem with Time Windows* (VRPTW). VRPTW merupakan salah satu pengembangan dari VRP dengan kendala tambahan berupa adanya *time windows* pada masing-masing tujuan. *Time windows* adalah batas waktu ketersediaan pelanggan, dimana waktu ketersediaan pada setiap pelanggan dapat berbeda-beda dan dinyatakan dalam selang waktu berupa batas waktu awal sampai waktu akhir pelayanan (Suprayogi & Mahmudy, 2015) Terdapat sejumlah kendaraan yang akan mendistribusikan barang ke sejumlah pelanggan. Setiap kendaraan berangkat dari tempat awal, yang disebut depot, mengunjungi sejumlah pelanggan, dengan setiap pelanggan memiliki batas waktu pelayanan, kemudian kembali lagi ke depot. Tujuan penyelesaian VRPTW adalah menentukan rute setiap kendaraan dari depot ke pelanggan dan meminimumkan total biaya perjalanan semua kendaraan. Untuk menyelesaikan VRPTW diperlukan metode penyelesaian yang tepat agar mendapatkan solusi optimal dengan waktu yang cepat.

Sub bab selanjutnya akan membahas model optimasi dari VRPTW. Model tersebut dibangun untuk meminimumkan total biaya perjalanan bagi semua kendaraan, dengan setiap kendalanya menentukan batasan-batasan yang harus dipenuhi pada VRPTW. Selanjutnya model tersebut akan diselesaikan dengan algoritma *Honey Bee Mating Optimization* (HBMO). HBMO adalah algoritma metaheuristik yang terinspirasi dari proses perkawinan lebah madu.

3.2 Model Optimisasi

Bagian ini membahas tentang model optimisasi VRPTW. Untuk kepentingan pemodelan terlebih dahulu akan di definisikan himpunan, parameter, dan variable keputusan sebagai berikut:

1. Himpunan

   V adalah himpunan pelanggan dan depot

   K adalah himpunan kendaraan

2. Parameter

   c_{ijk} adalah biaya dari pelanggan *i* ke pelanggan *j* menggunakan kendaraan *k*

C adalah kapasitas kendaraan

t_{ij} adalah waktu tempuh dari pelanggan *i* ke pelanggan *j*

a_i adalah waktu awal pelayanan pelanggan *i*

b_i adalah waktu akhir pelayanan pelanggan *i*

d_i adalah jumlah permintaan pelanggan *i*

s_{ik} adalah waktu kendaraan *k* mulai melayani pelanggan *i*

3. Variabel Keputusan

$$x_{ijk} = \begin{cases} 1, jika\ terdapat\ perjalanan\ dari\ i\ ke\ j\ dengan\ kendaraan\ k \\ 0, yang\ lainnya \end{cases}$$

Fungsi tujuan pada penelitian ini adalah untuk meminimumkan total biaya perjalanan. Fungsi ini dituliskan sebagai persamaan berikut:

$$z = \sum_{k \in \mathrm{K}} \sum_{j \in \mathrm{V}} \sum_{i \in \mathrm{V}} c_{ijk} x_{ijk}$$

Adapun kendala – kendala dari model VRPTW adalah sebagai berikut:ko

1. Setiap pelanggan hanya dikunjungi satu kendaraan. Kendala ini dinyatakan sebagai:

$$\sum_{i \in V} \sum_{j \in V} x_{ijk} = 1, \forall k \in K, V/\{0\}$$

2. Total permintaan dari tiap pelanggan tidak boleh melebihi kapasitas kendaraan. Kendala ini diekspresikan sebagai:

$$\sum_{i \in V} d_i \sum_{j \in V} x_{ijk} \leq C, \forall k \in K$$

3. Setiap kendaraan meninggalkan pelanggan yang telah dikunjungi. Ekspresi matematika dari kendala ini adalah:

$$\sum_{i \in V} x_{ilk} - \sum_{j \in V} x_{ljk} = 0, l \in V, \forall k \in K$$

4. Setiap kendaraan yang meninggalkan depot harus kembali ke depot. Kendala ini dinyatakan sebagai persamaan:

$$\sum_{j \in V} x_{0jk} = 1, \forall k \in K$$

$$\sum_{j \in V} x_{j0k} = 1, \forall k \in K$$

5. Kendaraan *k* tidak bisa mendatangi pelanggan *j* sebelum batas waktu yang ditentukan. Kendala ini diekspresikan sebagai:

$$s_{ik} + t_{ij} - L_{ij}(1 - x_{ijk}) \leq s_{jk}, \forall i, j \in V, \forall k \in K$$

dengan: $L_{ij} = b_i - a_j$

6. Kendaraan hanya bisa berada di pelanggan pada batas waktu yang ditentukan. Kendala ini dituliskan sebagai:

$$a_i \leq s_{ik} \leq b_i, \forall i \in V, \forall k \in K$$

Adapun batasan variable dari model optimasi VRPTW adalah:

$$x_{ijk} \in [0,1], \forall i \in V, \forall j \in V, \forall k \in K$$

Selengkapnya, model optimisasi dari VRPTW dituliskan sebagai berikut:

Meminimumkan:

$$z = \sum_{k \in K} \sum_{j \in V} \sum_{i \in V} c_{ijk} x_{ijk} \quad (3.1)$$

Berdasar:

$$\sum_{i \in V} \sum_{j \in V} x_{ijk} = 1, \forall k \in K \quad (3.2)$$

$$\sum_{i \in V} d_i \sum_{j \in V} x_{ijk} \leq C, \forall k \in K \quad (3.3)$$

$$\sum_{i \in V} x_{ilk} - \sum_{j \in V} x_{ljk} = 0, l \in V, \forall k \in K \quad (3.4)$$

$$\sum_{j \in V} x_{0jk} = 1, \forall k \in K \quad (3.5)$$

$$\sum_{j \in V} x_{j0k} = 1, \forall k \in K \quad (3.6)$$

$$s_{ik} + t_{ij} - L_{ij}(1 - x_{ijk}) \leq s_{jk}, \forall i, j \in V, \forall k \in K \quad (3.7)$$

$$a_i \leq s_{ik} \leq b_i, \forall i \in V, \forall k \in K \quad (3.8)$$

$$x_{ijk} \in [0,1], \forall i \in V, \forall j \in V, \forall k \in K \quad (3.9)$$

Model di atas termasuk dalam kategori model integer programming.

3.3 Implementasi Algoritma HBMO Pada Masalah VRPTW

VRPTW merupakan suatu permasalahan yang diklasifikasikan sebagai *NP-hard problem.* Ini berarti bahwa solusi *feasible* dari masalah tersebut sangat banyak dan

membutuhkan waktu yang lama untuk menemukan solusi optimalnya. Oleh karena itu, metode – metode penyelesaian alternatif dibutuhkan agar solusi optimal dapat diperoleh dengan cepat.

Pada penelitian ini, model VRPTW akan diselesaikan menggunakan algoritma *Honey Bee Mating Optimization* (HBMO). HBMO adalah salah satu algoritma yang telah terbukti mampu menyelesaikan permasalahan yang masuk dalam klasifikasi *NP-hard Problem.* Pada HBMO, solusi VRPTW direpresentasikan dengan kromosom. Sebuah kromosom terdiri dari sekumpulan gen yang merepresentasikan pelanggan. Pada tahap awal, kromosom-kromosom ini dibangkitkan secara acak dalam sebuah populasi. Kromosom – kromosom dalam suatu populasi dianalogikan sebagai himpunan-himpunan solusi VRPTW. Kemudian himpunan-himpunan solusi dievaluasi tingkat kesesuaiannya terhadap permasalahan yang ingin diselesaikan dengan sebuah nilai yang dinamakan nilai *fitness*. Nilai *fitness* ini biasanya berhubungan erat dengan permasalahan yang dibahas. Semakin besar nilai *fitness* maka himpunan solusi tersebut akan memiliki peluang yang besar untuk menuju solusi optimum. Nilai *fitness* yang tinggi akan memberikan kesempatan sebuah solusi terpilih sebagai solusi terbaik yang akan menjadi salah satu induk pada tahap reproduksi. Sedangkan induk yang lain akan terpilih lewat sebuah proses *flight mating,* yang juga memilih induk berdasarkan solusi yang memenuhi ketentuan. Kemudian kedua induk akan melakukan proses perkawinan silang yang akan menghasilkan anak sebagai solusi baru, yang membawa beberapa sifat dari induknya. Nantinya anak tersebut akan kembali diperiksa kesesuaiannya menggunakan nilai *fitness.* Jika nilai *fitness*-nya tidak lebih baik dari induknya maka solusi tersebut tidak akan diterima, dan induk yang sebelumnya merupakan solusi terbaik akan melakukan proses perkawinan dengan induk lain yang terpilih melalui proses *flight mating*. Dengan melalui semua proses algoritma HBMO, diharapkan menghasilkan solusi baru dengan nilai *fitness* yang lebih tinggi dibandingkan solusi sebelumnya. Setelah dilakukan dalam beberapa iterasi yang telah ditetapkan, maka akan dihasilkan solusi optimal (mendekati optimal).

3.2.1 Representasi Kromosom

Pada penelitian ini, kromosom direpresentasikan dalam bentuk *permutation encoding,* dimana gen menyatakan lokasi pelanggan dan panjang kromosom menyatakan banyaknya pelanggan. Setiap kendaraan (K) akan mengunjungi pelanggan untuk memenuhi permintaan dari pelanggan tersebut. Satu kromosom akan menghasilkan rute bagi setiap kendaraan.

Untuk mendapatkan rute tersebut dilakukan langkah – langkah sebagai berikut. Misal gen ke-i dari suatu kromosom dinyatakan dengan g_i.

1. Inisiasi rute pertama dari kendaraan pertama (K_1) dengan memasukan gen urutan pertama, menjadi $K_1 = \{g_1\}$.
2. Hitung total permintaan gen di K_1.
3. Jika total permintaan gen kurang atau sama dengan kapasitas $K_i, i = 1,2,3, \ldots$ maka tambahkan gen urutan selanjutnya g_{i+1}, jika tidak maka kurangi gen urutan terakhir dari K_1 dan masukan gen tersebut ke dalam rute baru K_{i+1}.
4. Lakukan langkah 3 sampai semua gen dalam satu kromosom sudah terpilih.

Kromosom	5	3	7	1	2	4	8	6	9

Gambar 3.1 Reperesentasi Kromosom

Gambar 3.1 mengilustrasikan contoh representasi kromosom dari 9 pelanggan, yaitu pelanggan 1,2,3,…,9. Misal kapasitas setiap kendaraan adalah 25. Maka untuk menentukan rute setiap kendaraan dilakukan langkah 1 sampai langkah 4 seperti di atas. Untuk rute pertama masukan urutan pelanggan pertama adalah 5 karena permintaannya 5<25, maka tambahkan pelanggan urutan selanjutnya dalam rute tersebut. Setelah langkah 3 diulang, rute pertama berisikan rute pelanggan 5-3-7-1-2-4 dengan total permintaan 22. Urutan pelanggan selanjutnya yaitu 8 tidak bisa ditambahkan ke dalam rute pertama karena jika ditambahkan membuat total permintaan melebihi kapasitas kendaraan sehingga dimasukan ke dalam rute kedua. Untuk rute kedua ulangi lagi langkah 3 sehingga rute kedua berisikan rute pelanggan 8-6-9 dengan total permintaan 23. Semua pelanggan dalam satu kromosom sudah terpilih sehingga satu kromosom menghasilkan rute pertama dan rute kedua. Setiap kendaraan akan memulai dan mengakhiri perjalanan dari depot yang dimisalkan dengan nilai 0. Jadi rute pertama untuk kendaraan ke-1 adalah 0-5-3-7-1-2-4-0 dengan total permintaan 22 dan rute kedua untuk kendaraan ke-2 adalah 0-8-6-9-0 dengan total permintaan 23.

3.2.2 Pembangkitan Populasi

Proses pembangkitan populasi dilakukan secara acak. Populasi berisi sejumlah kromosom yang telah direpresentasikan sebelumnya. Banyaknya kromosom yang dibangkitkan adalah sebanyak P_{rate} yang ditentukan. Misal pada penelitian ini P_{rate}-nya adalah 5, maka kromosom yang dibangkitkan yaitu sejumlah 5 kromosom. Kromosom dibangkitkan dengan langkah-langkah yang sudah dijelaskan sebelumnya dan misalkan dihasilkan populasi sebagai berikut:

D_1	1	2	5	3	4
D_2	4	1	2	5	3
D_3	2	3	1	4	5
D_4	5	2	4	3	1
D_5	2	4	1	3	5

dengan D_1 menyatakan kromosom 1 dan seterusnya.

3.2.3 Menghitung Nilai *Fitness*

Fitness adalah nilai yang ada pada tiap kromosom yang menentukan kesesuaian kromosom tersebut terhadap suatu permasalahan yang ingin diselesaikan (Mahmudy, dkk., 2014). Formula untuk menghitung nilai *fitness* disebut fungsi *fitness.* Tujuan dari penelitian ini adalah meminimumkan total biaya perjalanan semua kendaraan maka fungsi *fitness* yang digunakan adalah sebagai berikut:

$$F = \frac{1}{(z + total\ idle\ time + total\ waktu\ telat)}$$

dengan:

F = fungsi *fitness*

z = fungsi tujuan VRPTW

Misal kromosom-kromosom pada populasi sebelumnya akan dihitung nilai *fitness*nya. Sebelumnya kita tentukan fungsi objektif dari masing-masing kromosom. Misalkan nilai fungsi objektif dari masing-masing kromosom adalah sebagai berikut:

$z(D_1)$	21000
$z(D_2)$	21300
$z(D_3)$	21700
$z(D_4)$	26300
$z(D_5)$	24200

Maka diperoleh nilai *fitness* sebagai berikut:

$f(D_1)$	4.762E-05
$f(D_2)$	4.695E-05
$f(D_3)$	4.608E-05
$f(D_4)$	3.802E-05
$f(D_5)$	4.132E-05

3.2.4 Seleksi

Seleksi merupakan proses yang dilakukan setelah menghitung nilai *fitness.* Seleksi dilakukan untuk menentukan kromosom mana yang akan dipilih sebagai ratu. Pada penelitian ini, digunakan metode seleksi *roulette wheel.* Pemilihan kromosom terbaik dilakukan dengan cara menghitung nilai *fitness* kromosom kemudian membandingkannya dengan nilai *fitness* kromosom lainnya. Semakin besar nilai *fitness*nya, kemungkinan terpilihnya juga semakin besar.

Penelitian ini akan menggunakan metode seleksi *roulette wheel,* sehingga untuk menyeleksi kromosom akan dilakukan tahap - tahap metode seleksi *roulette wheel* sebagai berikut:

1. Menghitung nilai *fitness* individu dan menentukan total *fitness,* dengan rumus sebagai berikut:

$$Total\ nilai\ fitness = \sum_{k=1}^{5} f_k$$

Untuk contoh di atas diperoleh total nilai fitness sebagai berikut:

$$Total\ nilai\ fitness = \sum_{k=1}^{5} f_k = 0.00022$$

2. Menghitung nilai *fitness relatif* tiap individu yang dinotasikan dengan p_k dan didefinisikan sebagai berikut:

$$p_k = \frac{f_k}{Total\ nilai\ fitness}$$

Untuk contoh di atas diperoleh total nilai fitness relatif sebagai berikut:

p_1	0.2164547
p_2	0.213406
p_3	0.2094723
p_4	0.1728345
p_5	0.1878326

3. Menghitung nilai *fitness* kumulatif yang dinotasikan dengan q_k dan didefinikan sebagai berikut:

$$q_1 = p_1$$

$$q_k = q_{k-1}, k = 1,2,3,4,5$$

Untuk contoh di atas diperoleh total nilai fitness komulatif sebagai berikut:

C_1	0.23768
C_2	0.451086
C_3	0.6605583
C_4	0.8333928
C_5	1.0212253

4. Memilih *parent* yang akan menjadi calon untuk melakukan *crossover* dengan cara membangkitkan bilangan random r. Jika $q_k \in r$ dan $q_{k+1} > r$, maka pilih kromosom ke $k+1$ sebagai calon induk. Misalkan untuk contoh diatas r_1 = 0,252. Karena C_1 < 0,252 < C_2, maka kromosom 1 adalah D_2 Kromosom 2 = D_2. Selanjutnya misal $r_2 = 0{,}343$. Karena C_1 < 0,343 < C_2, maka kromosom 2 adalah D_2. Sehingga kromosom yang didapat adalah sebagai berikut:

   Kromosom 1 = D_2

   Kromosom 2 = D_2

   Kromosom 3 = D_5

   Kromosom 4 = D_5

   Kromosom 5 = D_3

Namun karena pada algoritma ini hanya dipilih satu kromosom terbaik sebagai ratu, maka akan dipilih kromosom yang muncul terbanyak yaitu D_2, dan kromosom yang tersisa akan bertindak sebagai *drone*.

3.2.5 *Flight Mating*

Setelah menentukan ratu, dilakukan proses *flight mating* dengan cara memilih *drone* secara acak, kemudian menghitung nilai probabilistiknya. Sebuah nilai acak diproduksi, kemudian *drone* akan melewati probabilistik aturan keputusan jika nilai fungsi

probabilistiknya lebih besar dari nilai acak. *Drone* yang melewati probabilistik aturan keputusan, *sperma*nya akan disimpan dalam *spermatheca* ratu. Proses ini berhenti jika *spermatheca* penuh atau ketika energi dan kecepatan ratu mencapai batas bawah. Kapasitas sperma berdasarkan nilai M yang ditentukan dan batas bawah energi dan kecepatan ratu berdasarkan nilai $S(t_{min})$ dan $E(t_{min})$ yang ditentukan.

Misal kita pilih sebuah *drone* secara acak dari populasi *drone* sebelumnya, dan yang terpilih adalah D_5. Kemudian kita hitung probabilitas D_5 dengan rumus sebagai berikut:

$$prob(Q, D_i) = \exp\left(\frac{-\Delta(f)}{S(t)}\right)$$

dengan:

$\Delta(f) = 2.59E - 06$

$S(1) = 1$

Untuk contoh di atas diperoleh:

$$prob(Q, D_5) = 0.999997$$

Selanjutnya *update* kekuatan dan energi ratu dengan rumus sebagai berikut:

$$S(t+1) = \alpha.S(t)$$
$$E(t+1) = \alpha.E(t)$$

Untuk contoh di atas diperoleh:

$S(2) = 0.98$

$E(2) = 0.98$

Langkah selanjutnya adalah membangkitkan bilangan acak r. Misal r = 0,172, maka untuk contoh di atas karena nilai $prob(Q, D_5)$ = 0,999 > r, sperma dari kromosom D_5 dimasukkan ke dalam *spermatheca* ratu, atau secara matematisnya kita notasikan $D_5 \rightarrow S_1$. Dengan demikian *spermatheca* ratu memuat satu kromosom. Misal pada penelitian ini kapasitas *spermatheca* ratu adalah 2, artinya *spermatheca* ratu belum penuh, sehingga proses *flight mating* dilakukan lagi dari proses pemilihan *drone* secara acak.

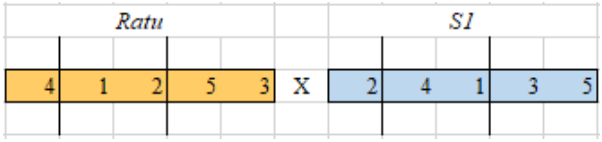
Setelah proses *flight mating* yang ke-dua dilakukan, diperoleh bahwa kromosom yang terpilih yaitu D_3, artinya $D_3 \rightarrow S_2$. Karena *spermatheca* sudah penuh, diperoleh kromosom yang akan masuk ke proses *crossover* sebagai berikut:

Ratu	Q	4	1	2	5	3
Sperma	S_1	2	4	1	3	5
	S_2	2	3	1	4	5

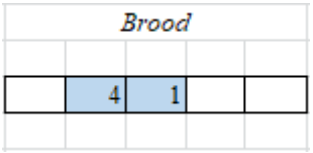
3.2.6 *Crossover*

Crossover merupakan operator algoritma HBMO yang digunakan untuk menggabungkan gen ratu dengan *drone*. Metode *crossover* yang digunakan dalam penelitian ini adalah *Order Crossover Operator*. Contoh langkah-langkah *Order Crossover Operator* adalah sebagai berikut:

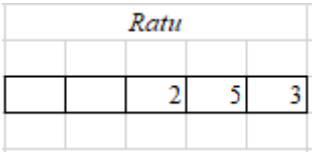
1. Menentukan dua titik *crossover* pada *parent*, dengan *parent* 1 adalah Ratu dan *parent* 2 adalah S_1.



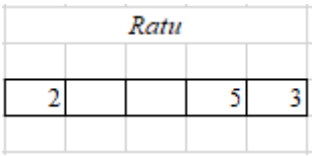
2. Mewariskan gen yang diapit di S_1 pada kromosom *child*, yang pada algoritma ini disebut *brood*.



3. Menentukan gen pada Ratu yang belum dipilih.



4. Menempatkan gen Ratu yang belum terpilih pada kromosom *brood*.



5. Menggabungkan kromosom pada langkah 2 dan 4

Brood				
2	4	1	5	3

3.2.7 Mutasi

Mutasi berfungsi untuk meningkatkan variasi populasi, menggantikan gen yang hilang selama proses seleksi serta menyediakan gen yang tidak ada dalam populasi awal. Pada penelitian ini akan dilakukan teknik *swapping mutation*, yaitu menukar posisi gen yang satu dengan yang lain. Sebelumnya dibangkitkan sebuah bilangan acak antara 0 sampai 1. Kemudian bilangan acak tersebut dibandingkan dengan *mutation_rate*. Jika nilai bilangan acak lebih besar dari nilai *mutation_rate* maka *brood* akan mengalami mutasi. Selanjutnya akan dibangkitkan dua bilangan acak yang menunjukkan posisi gen yang akan ditukar posisinya

Misalkan akan dilakukan mutasi pada *brood* hasil *crossover* sebelumnya. Missal kita pilih nilai *mutation_rate* sebesar 0,1. Kemudian bilangan acak yang dibangkitkan berniali 0,2 artinya bilangan acak tersebut lebih besar dari *mutation_rate* sehingga *brood* akan mengalami mutase. Selanjutnya dibangkitkan dua bilangan acak, dan diperoleh bilangan 2 dan 5, maka tukar posis gen 2 dan gen 5, sehingga diperoleh *brood* hasil mutasi sebagai berikut:

Brood	2	3	1	5	4

3.2.8 Pemilihan Ratu Baru

Setelah dilakukan proses mutasi, kita tentukan ratu untuk iterasi selanjutnya. Jika nilai *fitness* individu baru lebih baik dari nilai *fitness* ratu sebelumnya, maka ganti ratu sebelumnya dengan individu baru. Namun jika nilai *fitness* ratu sebelumnya lebih baik, ratu tidak akan diganti. Sebaliknya, nilai *fitness* individu baru akan dibandingkan dengan nilai *fitness drone.* Jika nilai *fitness* individu baru lebih baik dari salah satu *drone*, maka *drone* tersebut akan diganti dengan individu baru.

Untuk contoh di atas dihitung nilai *fitness*nya, diperoleh bahwa nilai *fitness brood* tidak lebih baik dari ratu. Maka kita bandingkan nilai *fitness brood* dengan nilai *fitness drone*. Ternyata diperoleh bahwa nilai *fitness brood* lebih baik dari nilai *fitness* D_4. Maka kita tukar D_4 dengan *brood* sebagai berikut:

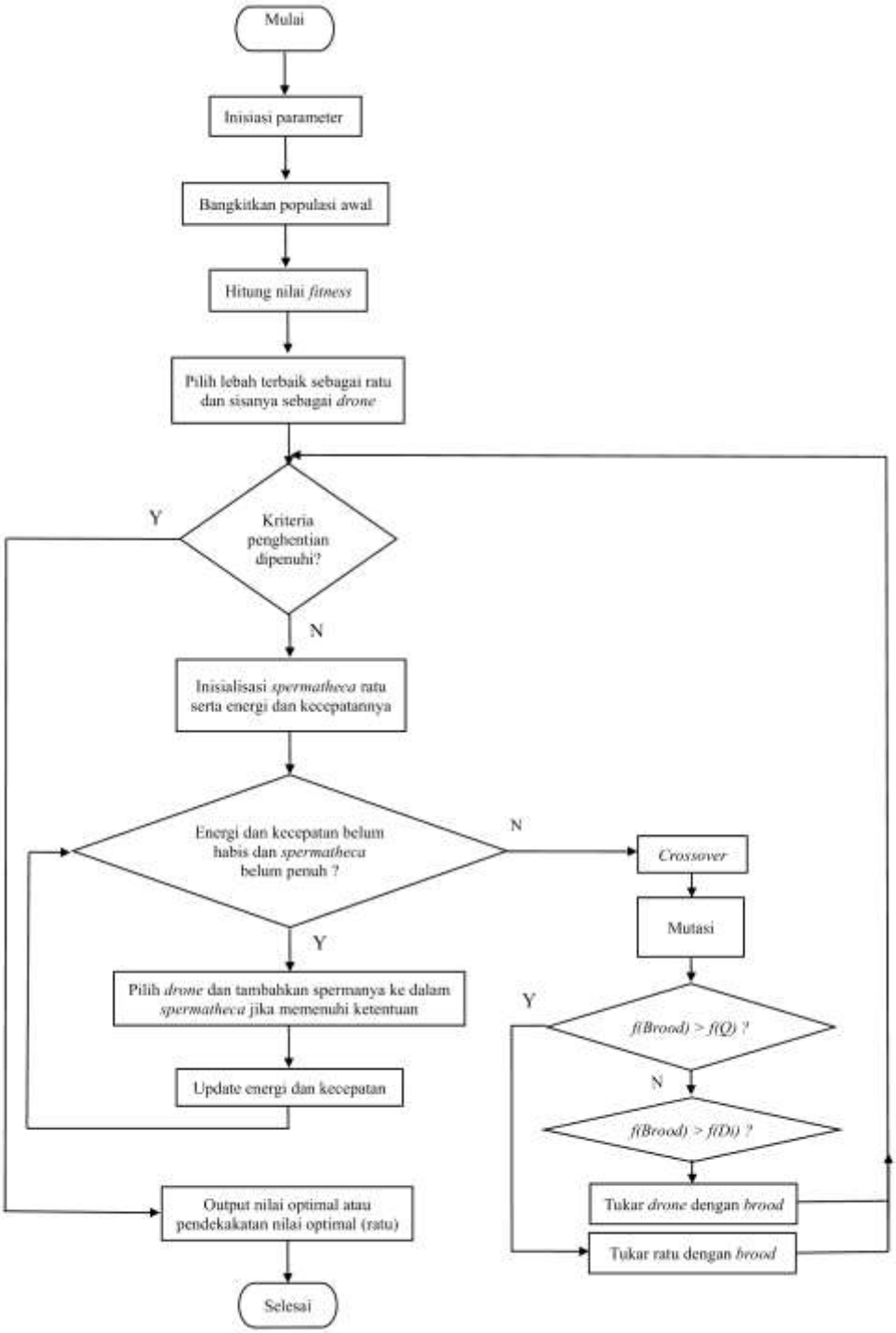
Ratu	Q	4	1	2	5	3
Drone	D_1	1	2	5	3	4
	D_3	2	3	1	4	5
	D_4	2	3	1	5	4
	D_5	3	4	5	1	2

3.2.9 Iterasi

Dengan diperolehnya populasi baru, proses algoritma HBMO telah mencapai satu iterasi. Proses dilanjutkan dari awal dengan kembali melakukan tahap *flight mating, crossover,* dan mutasi, kemudian memilih ratu baru untuk iterasi selanjutnya. Proses ini dilakukan hingga mencapai batas maksimum iterasi. Batas maksimum iterasi didasarkan dari nilai F_{max} yang ditentukan. Untuk memperjelas cara kerja algoritma HBMO, maka digambarkan cara kerjanya dalam bentuk *flowchart* pada Gambar 3.2.

3.4 Validasi

Validasi model digunakan untuk menguji apakah model yang sudah dibangun dan teknik penyelesaian yang dipilih sudah tepat atau tidak. Validasi dilakukan menggunakan data berukuran kecil yang memungkinkan untuk diselesaikan secara manual. Selanjutnya, solusi hasil perhitungan manual dibandingkan dengan solusi hasil teknik penyelesaian model. Jika diperoleh perbedaan, maka tahapan diulang dari tahap pembangunan model. Jika solusinya sama, maka dilanjutkan ke tahapan penelitian selanjutnya yaitu implementasi.



Gambar 3.2 *Flowchart* Algoritma HBMO untuk Permasalahan VRPTW