BAB III
METODOLOGI PENELITIAN

Bab ini membahas metodologi yang digunakan untuk menyelesaikan masalah penjadwalan kasus bedah yang diawali dengan mendeskripsikan masalah, menjelaskan tahapan penelitian, formulasi model optimisasi, dan teknik penyelesaiannya dengan menggunakan *Improved Multi-Objective Imperialist Competitive Algorithm* (IMOICA).

3.1 Deskripsi Masalah

Penelitian ini membahas masalah penjadwalan kasus bedah pada pasien bedah elektif (tidak *emergency*) yang meliputi tiga tahap, yaitu *pre-operative, peri-operative,* dan *post*-operative. Setiap pasien mempunyai tahapan pembedahan yang berbeda-beda. Pada setiap tahap, setiap pasien memerlukan serangkaian sumber daya bedah, yang meliputi dokter bedah, anestesiolog, perawat, ruang operasi dan ruang PACU.

Misalkan terdapat *m* pasien bedah. Setiap proses pembedahan pasien harus melalui *n* tahapan. Terdapat *k* buah sumber daya bedah. Setiap tahapan memerlukan waktu penyelesaian tertentu dan penggunaan sumber daya memerlukan biaya tertentu. Masalah penjadwalan kasus bedah adalah masalah memasangkan pasien terhadap tahapan pembedahan dan sumber daya bedah yang harus disesuaikan dengan waktu operasional rumah sakit.

Penelitian ini memodelkan masalah penjadwalan kasus bedah sebagai FJSP dengan mengasumsikan pasien adalah pekerjaan, sumber daya terkait sebagai mesin, dan tahapan operasi sebagai operasi. Tujuan penyelesaian model adalah untuk meminimumkan waktu penyelesaian seluruh tahapan pembedahan pasien (*makespan*) dan meminimumkan total biaya medis (TMC). Dengan demikian, masalah pembedahan tersebut dapat diklasifikasikan sebagai masalah optimisasi multi-objektif. Penelitian ini akan menyelesaikan masalah penjadwalan kasus bedah dengan menggunakan *Improved Multi-Objective Imperialist Competitive Algorithm* (IMOICA).

3.2 Tahapan Penelitian

Penelitian ini dilakukan melalui tahapan-tahapan berikut:

1. **Studi Pustaka**

   Pada tahap ini dilakukan studi pustaka dengan cara mempelajari konsep serta teori-teori mengenai masalah penjadwalan kasus bedah dan Metode IMOICA, bersumber pada berbagai literatur baik jurnal, buku, dan karya tulis lainnya.

2. **Pengumpulan Data**

   Data yang digunakan dalam penelitian ini terdiri dari data sumber daya bedah, yaitu dokter bedah, anestesiolog, perawat, ruang operasi, ruang PACU, data pasien, data waktu operasional rumah sakit, data jadwal dokter dan data biaya tahapan bedah tiap pasien.

3. **Pemodelan**

   Pada tahap ini akan diformulasikan model optimisasi dari masalah penjadwalan kasus bedah dengan terlebih dahulu mendefinisikan himpunan, parameter, dan variabel keputusan dari model optimisasi.

4. **Penyelesaian Model**

   Pada tahapan ini, model optimisasi dari masalah penjadwalan kasus bedah akan diselesaikan dengan menggunakan Metode IMOICA.

5. **Validasi**

   Pada tahap ini akan dilakukan validasi model dan teknik penyelesaiannya dengan cara membandingkan solusi optimal dari contoh kasus sederhana. Data sederhana dihitung secara manual, kemudian solusi optimalnya dibandingkan dengan perhitungan menggunakan *software*. Jika solusi keduanya sama, maka dilanjutkan ke tahap implementasi. Jika diperoleh solusi yang berbeda, maka tahapan akan diulang mulai dari pemodelan.

6. **Implementasi**

   Setelah model dan teknik penyelesaian valid, selanjutnya model dan teknik penyelesaian tersebut akan diimplementasikan pada penyelesaian masalah penjadwalan kasus bedah pada sebuah rumah sakit dengan menggunakan Metode IMOICA.

7. Penarikan Kesimpulan

Pada tahapan ini, dilakukan penarikan kesimpulan berdasarkan hasil implementasi penyelesaian masalah penjadwalan kasus bedah menggunakan Metode IMOICA.

3.3 Model Optimisasi

Model optimisasi dari masalah penjadwalan kasus bedah dalam penelitian ini dibangun dengan asumsi-asumsi sebagai berikut:

1. Pasien merupakan pasien untuk kasus bedah elektif.
2. Semua pasien sudah siap sebelum operasi.
3. Tidak ada operasi yang diprioritaskan.
4. Sumber daya bedah seperti anestesiolog dan perawat dapat digunakan untuk operasi apapun.
5. Kumpulan sumber daya bedah hanya dapat menangani satu tahap bedah dalam satu waktu.
6. Setelah operasi dilakukan, operasi tidak dapat dihentikan sampai selesai.
7. Pasien akan melanjutkan ke tahap berikutnya hanya jika tahap bedah sebelumnya telah selesai.
8. Waktu pemrosesan setiap tahapan dan biaya pembedahan ditentukan terlebih dahulu dan tidak akan berubah seiring dengan urutan.
9. Waktu pemrosesan tahapan *peri-operative* sudah termasuk waktu persiapan ruang operasi.

Model optimisasi akan formulasikan merujuk pada model optimisasi FJSP yang dirumuskan oleh Brucker (Brucker & Schlie, 1990). Langkah-langkah yang dilakukan untuk memformulasikan model optimisasi adalah sebagai berikut:

1. Mendefinisikan himpunan dan parameter model.

   Himpunan model terdiri dari himpunan pasien, tahapan bedah, dan sumber daya bedah. Sedangkan sebagai parameter model adalah waktu pemrosesan tahapan bedah, waktu perpindahan pasien, waktu persiapan pasien, biaya tahapan bedah, biaya perpindahan pasien dan biaya persiapan pasien.

2. Mendefinisikan variabel keputusan.

   Variabel keputusan model menentukan pemasangan sumber daya ke tahapan operasi.

3. Merumuskan fungsi tujuan.

   Fungsi tujuan dari model optimisasi terdiri dari:

   a. Meminimumkan total waktu penyelesaian seluruh tahapan pembedahan pasien.

      Total waktu pembedahan merupakan penjumlahan dari lamanya waktu pasien untuk menyelesaikan seluruh proses pembedahan yang terdiri dari tahapan *pre-operative, peri-operative, post-operative*, waktu persiapan dan waktu perpindahan.

   b. Meminimumkan total biaya pembedahan

      Total biaya pembedahan merupakan penjumlahan dari biaya biaya medis selama pembedahan, biaya medis dalam proses perpindahan pasien dan biaya medis dalam proses persiapan pasien.

4. Menurunkan fungsi kendala model.

   Kendala dari model optimisasi adalah sebagai berikut:

   a. Setiap tahap dalam pembedahan harus dialokasikan ke sumber daya yang tersedia.
   b. Sumber daya bedah yang digunakan setiap tahap pembedahan adalah sumber daya yang tersedia dan sesuai untuk tahap tahap tersebut.
   c. Jika suatu tahap operasi dilakukan di sumber daya tertentu, tahap berikutnya harus dilakukan di sumber daya yang sesuai.
   d. Setiap jenis sumber daya hanya boleh digunakan oleh satu tahap pembedahan dalam satu waktu agar tidak terjadi bentrokan atau tumpang tindih penggunaan.
   e. Pasien tidak boleh memulai tahap operasi berikutnya sebelum tahap sebelumnya selesai.
   f. Total waktu penyelesaian seluruh tahapan bedah merupakan waktu terlama yang dibutuhkan untuk menyelesaikan tahapan pembedahan pada seorang pasien.

5. Menentukan batasan variabel keputusan.

   Batasan variabel keputusan model optimisasi menjamin bahwa seluruh variabel bernilai non negatif.

3.4 Teknik Penyelesaian dengan IMOICA

Penelitian ini menggunakan IMOICA untuk menyelesaikan masalah penjadwalan kasus bedah. IMOICA merupakan pembaruan dari metode ICA untuk menyelesaikan masalah multi objektif dengan lebih baik. Beberapa istilah yang akan digunakan dalam metode IMOICA adalah sebagai berikut:

1. *Encoding* merupakan proses mengubah data pasien dan operasi ke dalam format yang bisa digunakan oleh algoritma.
2. Strategi asimilasi merupakan proses di mana koloni (solusi yang lebih lemah) bergerak mendekati penjajah (solusi yang lebih kuat) agar menjadi lebih baik.
3. *Attraction and Repulsion* (AR) dalam asimilasi adalah metode yang menentukan bagaimana koloni bergerak menuju penjajah dalam strategi asimilasi.
   - *Attraction* (daya tarik) yaitu jika koloni cukup dekat dengan penjajah, maka koloni akan lebih tertarik untuk bergerak mendekati penjajah agar menjadi lebih baik.
   - *Repulsion* (tolakan) yaitu jika koloni terlalu jauh atau terlalu dekat dengan penjajah, maka strategi mutasi atau persilangan diterapkan agar koloni tidak hanya mengikuti penjajah tetapi juga mencari solusi baru yang lebih baik.
4. Solusi pareto adalah kumpulan solusi terbaik yang tidak bisa diperbaiki dalam satu aspek tanpa mengorbankan aspek lain.
5. Strategi revolusi adalah cara untuk mengubah solusi secara drastis agar tidak terjebak dalam solusi yang kurang optimal.
6. Strategi *Variable Neighborhood Search* (VNS) yaitu metode eksplorasi solusi yang membantu algoritma menemukan jawaban yang lebih baik dengan menghindari jebakan solusi lokal.

Fitur utama dari IMOICA yaitu, strategi hirarki sosial dikembangkan untuk menginisialisasi kerajaan, untuk meningkatkan kemampuan pencarian global algoritma, konsep *Attraction and Repulsion* (AR) diperkenalkan ke dalam strategi asimilasi, strategi revolusi digunakan untuk keanekaragaman populasi dan strategi

Variable Neighborhood Search (VNS) digunakan untuk meningkatkan kapasitas eksploitasi algoritma (Yu dkk., 2022). Kerangka kerja IMOICA dijelaskan dalam Algoritma 1.

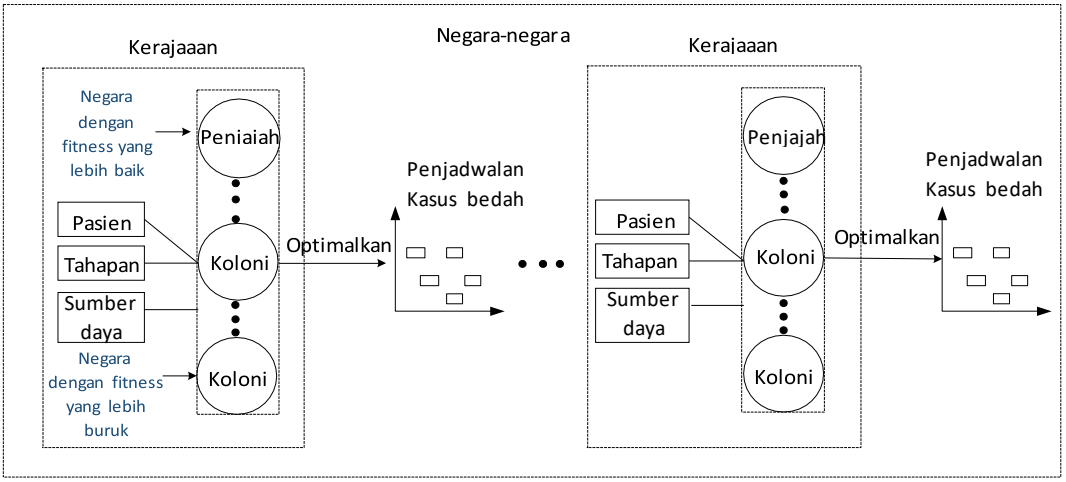
Algoritma 1. IMOICA

Input: semua populasi

Output: solusi optimal Pareto

Menghasilkan populasi awal secara acak
Inisialisasi kerajaan-kerajaan
For setiap penjajah *i* **do**
  Generasi solusi Pareto
  For setiap koloni *j* **do**
    Lakukan strategi asimilasi dengan ukuran AR
    Lakukan strategi pembaruan kerajaan
    Menerapkan strategi revolusi
    Jalankan strategi VNS
  End
End

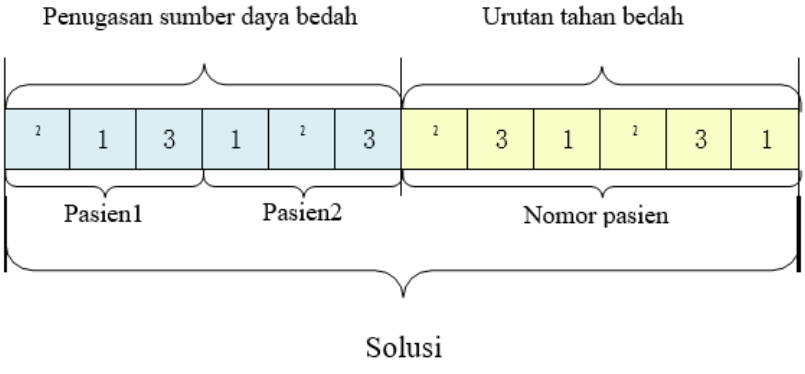
Dalam penelitian ini, negara setara dengan individu, semua negara (individu) merupakan populasi, dan mengoptimalkan negara (individu) setara dengan mengoptimalkan penjadwalan kasus bedah dari sekelompok pasien. Urutan penjadwalan diperoleh dengan algoritma optimisasi. Penerapan algoritma dalam penjadwalan kasus bedah ditunjukkan pada Gambar 3.1. Secara umum, tahapan IMOICA dalam menyelesaikan masalah penjadwalan kasus bedah terdiri dari *encoding*, inisialisasi kerajaan, membangkitkan solusi pareto, strategi asimilasi dengan pengukuran AR, memperbarui kerajaan, menerapkan strategi revolusi, dan jalankan strategi VNS. Sub bab berikut adalah penjelasan dari setiap tahapan.



Gambar 3.1 Penerapan Algoritma dalam Penjadwalan Kasus Bedah.

3.4.1 *Encoding*

Untuk *n* pasien yang perlu menjalani pembedahan, pengkodean didasarkan pada urutan tahapan pembedahan yang berbeda dan sumber daya yang berbeda yang dipilih. Pengkodean terdiri dari dua bagian, yaitu penugasan sumber daya bedah atau *Surgical Resource Assignment* (SRA), dan urutan tahap bedah atau *Surgical Stage Sequence* (SSS). Gambar 3.2 menunjukkan komposisi dari solusi pengkodean. SRA menyimpan nomor set sumber daya bedah yang tersedia yang dipilih untuk setiap tahap pembedahan. SSS mendefinisikan nomor pasien. Selanjutnya, strategi seleksi lokal diadopsi di bagian SRA dan strategi seleksi acak diadopsi di bagian SSS (Kazarlis dkk, 2015).



Gambar 3.2 Contoh Solusi *Encoding*.

3.4.2 Inisialisasi kerajaan

Strategi hirarki sosial digunakan untuk menginisialisasi kerajaan. Pertama, semua negara diurutkan berdasarkan *fitness*. Untuk menghitung nilai *fitness*, perlu mempertimbangkan beberapa aspek penting yang berkontribusi terhadap efisiensi penjadwalan kasus bedah. Nilai *fitness* terdiri dari:

1. Total waktu operasi, yaitu waktu yang dibutuhkan untuk menyelesaikan semua tahapan operasi (*pre-operative*, *peri-operative*, *post-operative*) untuk semua pasien. Total waktu operasi dituliskan sebagai:

$$TWO = \sum (\text{durasi } pre\_operative + \text{durasi } peri\_operative + \text{durasi } post\_operative$$

2. Efisiensi penggunaan sumber daya, yaitu seberapa baik sumber daya digunakan. Efisiensi ini dihitung sebagai jumlah total sumber daya yang digunakan dibandingkan dengan jumlah maksimal sumber daya yang tersedia dan dirumuskan sebagai berikut:

$$ESD = \frac{\text{sumber daya bedah yang digunakan}}{\text{sumber daya bedah yang tersedia}}$$

3. Waktu tunggu pasien, yaitu waktu tunggu pasien sebelum operasi dimulai atau selama perpindahan antar tahap operasi. Waktu tunggu tersebut dirumuskan sebagai:

$$WTP = \sum (\text{waktu tunggu pre\_operative} + \text{waktu tunggu peri\_operative} + \text{waktu tunggu post\_operative})$$

Nilai *fitness* merupakan kombinasi dari ketiga komponen di atas, dan rumus berikut:

$$Fitness = \mu_1 . TWO + \mu_2 . ESD + \mu_3 . WTP$$

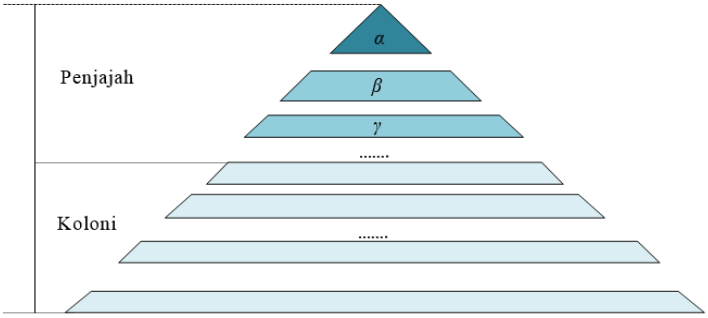
dengan μ_1, μ_2 dan μ_3 adalah bobot yang diberikan untuk masing masing komponen. Nilai *fitness* yang lebih rendah menunjukkan solusi yang lebih baik.

Nilai *Fitness* akan digunakan untuk menyusun hirarki sosial. Negara dengan nilai *fitness* terkecil menempati hirarki sosial tertinggi yang dinamakan penjajah α (lihat Gambar 3.3). Secara berurutan, level paling bawah dinamakan penjajah b, penjajah c, dan seterusnya. Negara-negara yang dengan nilai *fitness* terkecil disebut koloni. Koloni-koloni ini dibagi sesuai dengan hierarki sosial para penjajah, yaitu,

semakin tinggi hierarki sosial, semakin banyak koloni yang akan diduduki. Penduduk koloni oleh kaum penjajah bersifat acak. Oleh karena itu, setelah pemilihan penjajah, urutan koloni yang tersisa akan terganggu. Kemudian, para penjajah dan koloni akan membentuk kerajaan yang berbeda (Yu dkk, 2022). Jumlah koloni yang ditempati oleh para penjajah diperoleh dengan persamaan berikut:

$$Num(x) = \frac{Ncl}{(x+1)^2}$$

di mana $x = 1, \dots, Nim$; Nim adalah jumlah penjajah, $Num(x)$ adalah jumlah koloni yang diakuisisi oleh penjajah x, dan Ncl adalah jumlah koloni. Selama inisialisasi kerajaan, semakin kecil x, semakin tinggi hirarki sosial kerajaan tersebut.



Gambar 3.3 Hirarki Sosial Dibagi Berdasarkan Negara.

3.4.3 Membangkitkan solusi Pareto

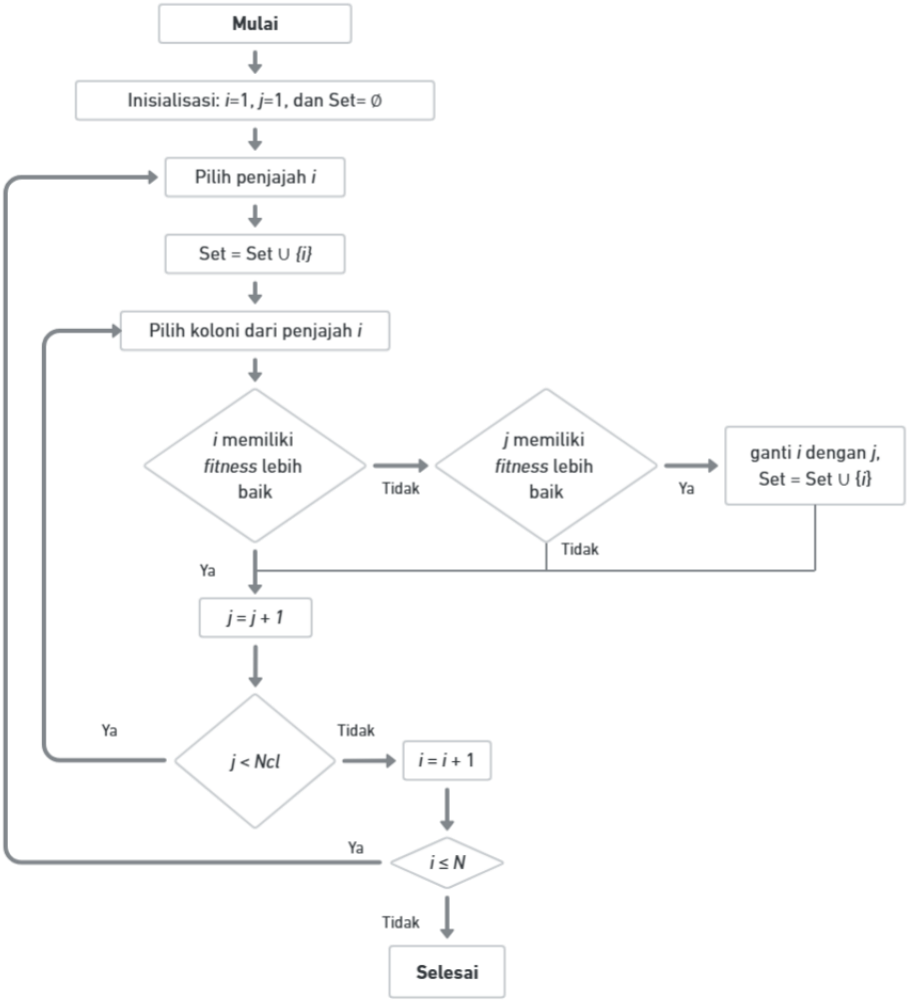
Dengan menggunakan strategi pengurutan non-dominan (Hansen & Mladenović, 2001), solusi optimum dalam populasi dipilih sebagai solusi Pareto. Populasi awal dan himpunan konstruksi masing-masing dilambangkan dengan P dan Ω. Individu-individu dalam Ω bersifat sementara, karena mereka dapat dihapus pada perbandingan berikutnya. Pada awal algoritma, individu pertama dimasukkan ke dalam himpunan konstruksi Ω, dan individu p' dalam populasi evolusioner $P(p' \notin \Omega)$ dihapus dan dimasukkan ke dalam himpunan konstruksi Ω. Kemudian, individu p' dibandingkan dengan individu dalam Ω secara bergantian, dan individu

yang didominasi oleh p' akan dihapus dari Ω. Tahapan untuk metode pengurutan non-dominan dijelaskan dalam Algoritma 2.

Pada iterasi pertama, ada Nim penjajah, masing-masing penjajah i memiliki $Ncli$ koloni. Oleh karena itu, strategi dimulai dengan penjajah-penjajah tersebut. Untuk setiap kerajaan, penjajah dipertahankan untuk membentuk solusi Pareto. Kemudian, penjajah tersebut dibandingkan dengan koloni-koloni yang tersisa. Penjajah tetap berada dalam kumpulan solusi Pareto selama penjajah tersebut mendominasi koloni-koloni; jika tidak, penjajah tersebut digantikan oleh koloni. Pada iterasi berikutnya, penjajah dari kerajaan-kerajaan tersebut dibandingkan dengan solusi individual dari solusi Pareto sebelumnya, dan solusi optimal Pareto diperoleh. Strategi ini ditunjukkan pada Gambar 3.4.

Algoritma 2. Pengurutan non-dominan (*non-dominated sorting)*

$\Omega = \text{find\_nondominated \_front(P)}$;
For setiap $p' \in P \wedge p' \notin \Omega$ **do**
  $\Omega = \Omega \cup \{p'\}$
  For setiap $q \in \Omega \wedge q \neq p'$ **do**
    If $p' \succ q$ **do**
      $\Omega = \Omega \setminus \{q\}$
    End
    Else $p' \prec q$ **do**
      $\Omega = \Omega \setminus \{p'\}$
    End
  End
End



Gambar 3.4 Tahapan Membangkitkan solusi Pareto.

3.4.4 Strategi asimilasi dengan pengukuran AR

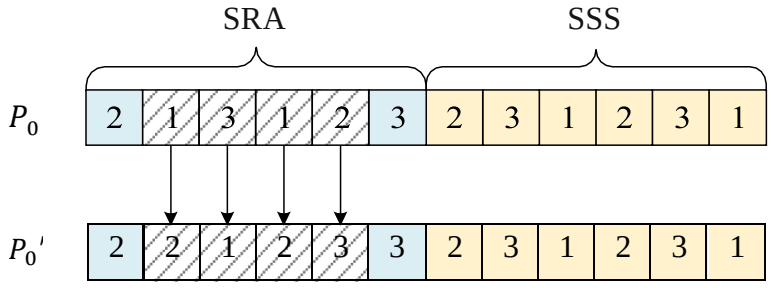
Faktor penting yang mempengaruhi strategi asimilasi adalah jarak antara penjajah dan koloni. Idenya adalah untuk menggunakan ukuran AR berdasarkan jarak ini untuk meningkatkan kinerja algoritma dan mencapai posisi optimal secara global. Jarak rata-rata antara penjajah dan koloninya (AVR) adalah sebagai berikut:

$$AVR = \frac{1}{M}\sum_{i=1}^{M}\left[\left(V_{best}^{col} - V^{Imp}\right) - \left(V_{current}^{col} - V^{Imp}\right)\right]$$

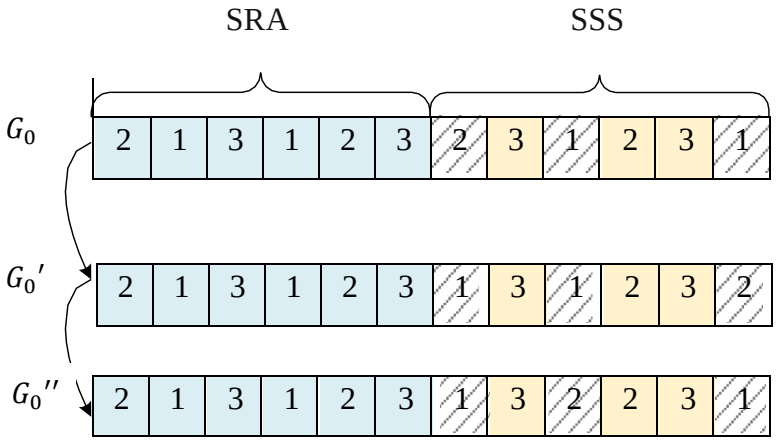
di mana M adalah jumlah koloni di setiap kerajaan, V_{best}^{col} adalah *fitness* dari koloni terbaik, $V_{current}^{col}$ adalah *fitness* dari koloni saat ini, dan V^{Imp} adalah *fitness* dari

penjajah. Setelah menghitung nilai AVR, tiga skenario berikut ini terjadi sesuai dengan nilai AVR:

1. Jika nilai AVR kurang dari ambang batas (ω) tertentu yang ditetapkan sebelumnya, operator mutasi dilakukan. Gambar 3.5 (a) menunjukkan operator mutasi yang terjadi pada bagian SRA. Pertama, individu yang layak (P_0) dipilih. Kemudian, beberapa elemen pada bagian SRA dipilih secara acak dan diganti dengan nomor set sumber daya bedah yang tersedia. Individu yang baru (P_0') yang baru dihasilkan. Gambar 3.5 (b) menunjukkan operator mutasi yang bekerja pada bagian SSS. Pertama, tiga titik variasi dipilih secara acak pada individu (G_0) sesuai dengan rentang pencarian lingkungan yang dihasilkan. Kemudian, semua kombinasi solusi lingkungan dibangkitkan (G_0', G_0''). Terakhir, salah satu dari solusi-solusi lingkungan dipilih secara acak sebagai keturunan.



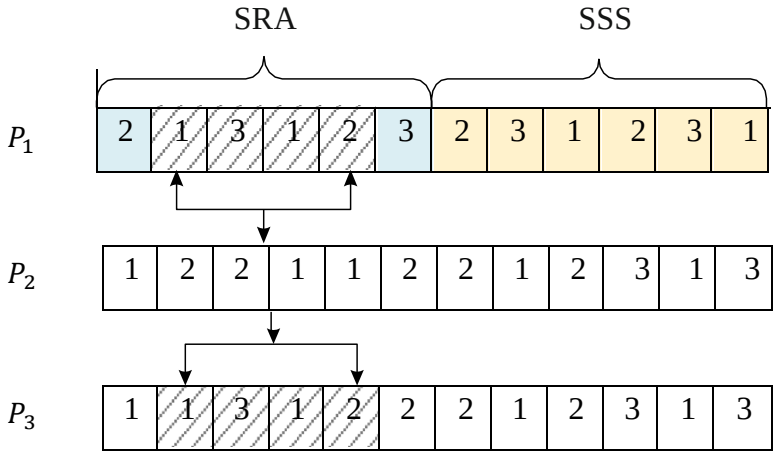
(a) Operator mutasi bagian SRA.



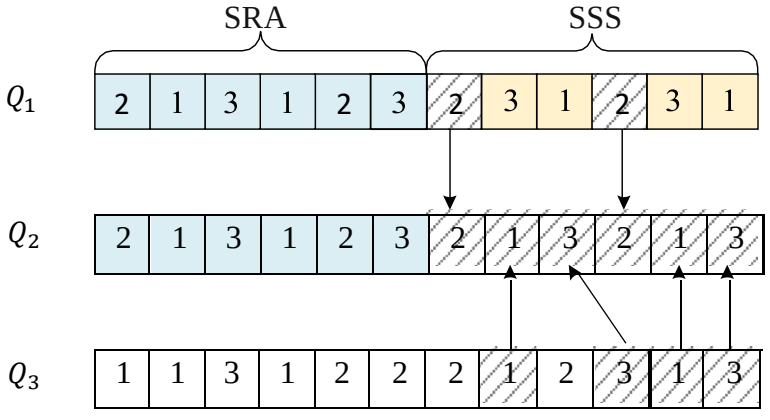
(b) Operator mutasi bagian SSS.

Gambar 3.5 Operator Mutasi.

2. Jika nilai AVR lebih besar dari ω, operator *crossover* dilakukan. Gambar 3.6 a) adalah strategi *crossover* dua titik yang bekerja pada bagian SRA. Pertama, dua posisi individu (P_1) dipilih secara acak. Kemudian, dengan memilih individu lain (P_2), elemen-elemen di antara dua posisi P_1 ini dipilih dan dimasukkan ke dalam P_2 untuk membentuk individu baru (P_3). Gambar 3.6 b) menunjukkan strategi *crossover* JBX yang bekerja pada bagian SSS. Pertama, nomor pasien dipilih secara acak dari individu (Q_1), dan menurut posisinya, nomor pasien dimasukkan ke dalam keturunannya (Q_2). Kemudian, individu lain (Q_3) dipilih, dan nomor pasien yang tersisa dimasukkan ke dalam Q_2 secara bergantian.



(a) Persilangan dua titik



(b) Persilangan JBX

Gambar 3.6 Operator Persilangan.

3. Jika nilai AVR sama dengan ω, maka strategi asimilasi klasik dilakukan. Proses asimilasi dapat dicapai melalui pergerakan koloni ke penjajah.

Kerangka kerja untuk strategi asimilasi dengan ukuran AR dijelaskan dalam Algoritma 3.

Algoritma 3. Strategi asimilasi dengan ukuran AR

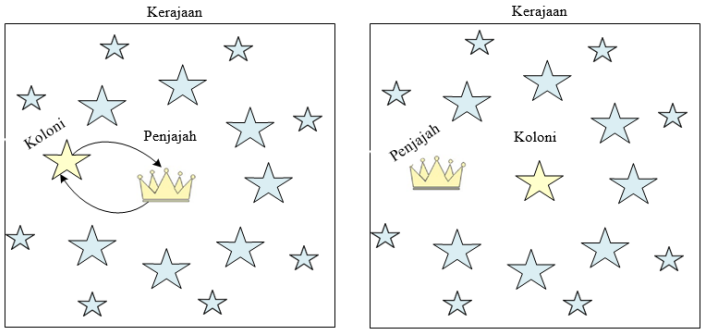
Hitung nilai AVR
Pilih ambang batas ω
For $AVR < \omega$ **do**
Operator mutasi dilakukan
End
For $AVR > \omega$ **do**
Operator persilangan dilakukan
End
Else
Strategi asimilasi klasik diimplementasikan
End

3.4.5 Memperbarui kerajaan

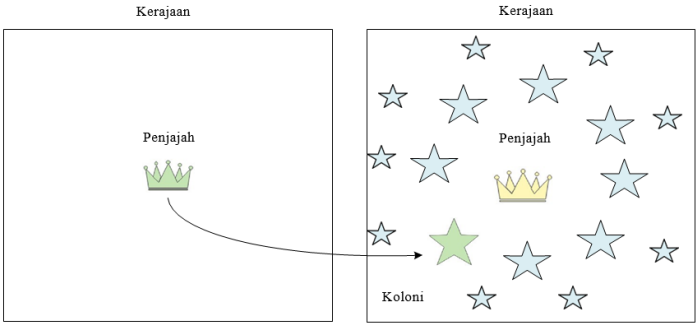
Pembaruan kerajaan mencakup dua situasi yaitu:

1. Penggantian penjajah: dengan bertambahnya iterasi, koloni-koloni dapat memperoleh kekuatan yang lebih besar daripada penjajah mereka dalam sebuah kerajaan. Akibatnya, kerajaan diganti, yaitu koloni yang paling kuat akan menggantikan penjajah sebagai penjajah baru, seperti yang digambarkan pada Gambar 3.7 (a).
2. Kepunahan kerajaan: untuk kerajaan yang tidak memiliki koloni, strategi kepunahan kerajaan diimplementasikan, yaitu untuk kerajaan yang hanya memiliki satu penjajah akan ditransfer ke kerajaan yang lebih kuat dan menjadi salah satu koloninya. Gambar 3.7 (b) menunjukkan situasi lain dalam pembaruan kerajaan.

Kerangka kerja untuk memperbarui kerajaan disajikan dalam Algoritma 4.



(a) Pembaruan penggantian



(b) Pembaruan kepunahan

Gambar 3.7 Memperbarui Kerajaan.

Algoritma 4. Memperbarui kerajaan

Jalankan strategi pembaruan pertama

For setiap kerajaan i **do**

IF koloni di kerajaan i lebih baik daripada penjajahnya **do**

Gantikan penjajah awal dengan penjajah baru

End

End

Jalankan strategi pembaruan kedua

For setiap kerajaan i **do**

IF tidak ada koloni di kerajaan i **do**

Strategi pemusnahan kerajaan diterapkan, yaitu, penjajah dipindahkan ke kerajaan yang kuat dan menjadi salah satu koloninya.

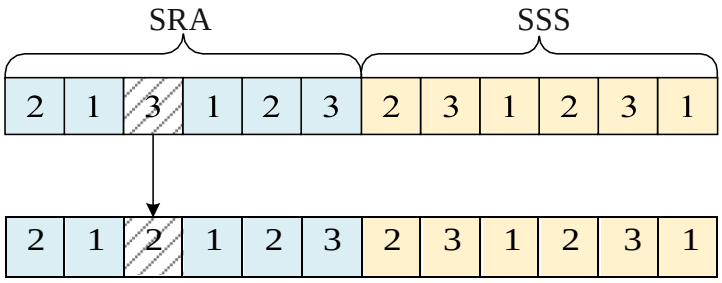
End

End

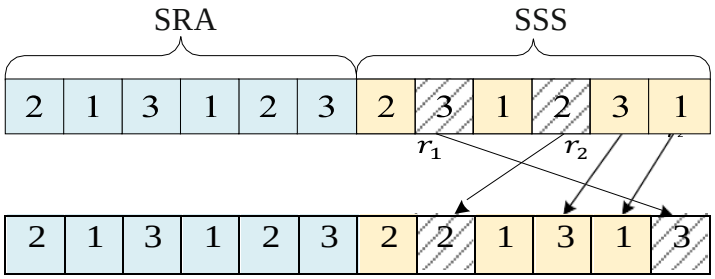
3.4.6 Menerapkan strategi revolusi

Revolusi adalah cara lain untuk menghasilkan solusi baru. Langkah-langkah rinci dari strategi revolusi tercantum dalam Algoritma 5, di mana R dan α adalah bilangan bulat, Pr adalah probabilitas revolusi (konsisten dengan parameter pada referensi), S_i adalah jumlah negara di kerajaan i, dan $rand$ adalah angka yang dihasilkan secara acak. Untuk koloni λ, U_λ mewakili jumlah koloni yang mendominasi koloni λ di dalam kerajaan. Koloni yang baik memiliki kemungkinan yang lebih besar untuk menjalankan strategi revolusi. Oleh karena itu, koloni λ dengan nilai U_λ terkecil akan memiliki kesempatan lebih besar untuk menjalankan strategi revolusi untuk menghasilkan koloni baru.

Operator pengubah dan penyisipan digunakan dalam strategi revolusi. Operator pengubah menghasilkan solusi baru dengan mengganti sebuah elemen secara acak pada bagian SRA, seperti yang ditunjukkan pada Gambar 3.8 (a). Operator penyisipan terdiri dari pemilihan dua elemen posisi r_1 dan r_2, secara acak di bagian SSS. Elemen r_2 disisipkan ke dalam r_1, dan elemen r_1 disisipkan ke dalam posisi terakhir. Kemudian, elemen setelah r_2 bergerak maju. Gambar 3.8 (b) menunjukkan sebuah contoh dari operator penyisipan.



(a) Operator Pengubah



(b) Operator penyisipan

Gambar 3.8 Operator Pengubah dan Penyisipan Pada Strategi Revolusi.

Algoritma 5. Revolusi

For $i = 1$ to N_{im} **do**
  $\alpha \leftarrow 1$
  For $j = 1$ to S_i **do**
    If $rand < P_r$ **do**
      $\alpha = \alpha + 1$
    End
  End
  Ubah Pareto *front* dengan membandingkan semua koloni di kerajaan
  If $\alpha > 1$ **do**
    For $\beta = 1$ to R **do**
      Pilih koloni λ dengan U_λ terkecil
      If $r = 1$ **do**
        Operator penyisipan dilakukan di koloni λ
      End
      Else
        Lakukan operator pengubah di koloni λ
      End
      Sebuah solusi baru SL_0 diperoleh
      If SL_0 didominasi λ **do**
        Koloni diganti
        For semua individu di Pareto *front* **do**
          If z didominasi m **do**
            Pareto *front* diperbarui
          End
        End
      End
    End
  End
End

3.4.7 Menjalankan strategi VNS

VNS adalah algoritma heuristik untuk memecahkan masalah optimisasi. VNS memiliki karakteristik struktur yang sederhana dan parameter yang sedikit, serta mekanisme lingkungan variabel yang unik dapat mencegah pencarian jatuh ke dalam optimisasi lokal. Hal ini memberikan VNS kemampuan pencarian lokal yang kuat . Dalam penelitian ini, strategi VNS ditambahkan untuk meningkatkan kinerja konvergensi algoritma. Langkah-langkah VNS tercantum dalam Algoritma 6, di mana tiga struktur tetangga yaitu penyisipan, pengubah, dan penukar digunakan. Operator pengubah dan penyisipan seperti yang dijelaskan di Bagian 3.4.6. Operator penukar bekerja pada bagian SSS, memilih dua elemen posisi secara acak dan kemudian menukarnya. max_0 adalah jumlah maksimum siklus yang dievaluasi oleh fungsi objektif untuk penghentian. Solusi-solusi yang tidak layak akan dibuang. Solusi layak yang dihasilkan dibandingkan dengan solusi saat ini, dan jika solusi layak lebih baik, solusi saat ini akan diganti.

Algoritma 6. Kerangka kerja VNS

Pilih anggota pertama dari Pareto *front* sebagai solusi saat ini x
$v, g \leftarrow 1$
While $v \leq$ max _0 **do**
If $g < 4$ **do**
$r = rand(\,)\%3$
If $r = 0$ **do**
Jalankan operator penukar
End
Else if $r = 1$ **do**
Jalankan operator penyisipan
End
Else
Jalankan operator pengubah
End
Hasilkan solusi baru z
If z didominasi x **do**

Perbarui Pareto *front*

Misalkan $g \leftarrow 1$

End

Else

$g \leftarrow g + 1$

End

$v \leftarrow v + 1$

If v v dinyatakan dengan bilangan bulat L_n **do**

Pilih secara acak solusi y dan bukan solusi x

End

End

End

Dalam metode IMOICA, tahapan asimilasi, revolusi, dan VNS tidak selalu diterapkan secara berurutan tanpa syarat sebab ketiga tahapan tersebut akan menghasilkan sebuah solusi baru. Terdapat kondisi tertentu yang menentukan apakah solusi akan melanjutkan ke tahap revolusi dan VNS setelah melalui asimilasi. Berikut adalah penjelasan kondisi yang memungkinkan transisi antara ketiga tahap tersebut:

1. Transisi dari asimilasi ke revolusi

   Jika hasil asimilasi tidak cukup meningkatkan kualitas solusi atau tidak ada perubahan signifikan dalam populasi, maka strategi revolusi diterapkan. Beberapa indikator yang menyebabkan tahapan asimilasi berlanjut ke tahapan revolusi yaitu:

   - Nilai AVR masih cukup besar setelah asimilasi, yang berarti solusi belum cukup dekat dengan optimal.
   - Tidak ada perubahan signifikan dalam Pareto front atau tidak ada solusi yang lebih baik setelah asimilasi.

2. Transisi dari Revolusi ke VNS

   Jika solusi baru yang dihasilkan dari revolusi tidak mendominasi solusi Pareto sebelumnya atau perbedaan *fitness* antara solusi terbaik dan solusi baru masih besar maka strategi VNS akan diterapkan.

3. Transisi langsung dari asimilasi ke VNS (tanpa revolusi)

   Dalam beberapa kasus, solusi dari tahap asimilasi mungkin sudah cukup baik dan tidak memerlukan revolusi. Jika AVR setelah asimilasi sudah cukup kecil, menunjukkan bahwa koloni sudah dekat dengan penjajah maka tahapan bisa langsung dilanjutkan ke VNS untuk mengoptimalkan solusi secara lokal.