

Bab 7

Algoritma Genetika

POKOK BAHASAN:

- ✓ Beberapa definisi Penting dalam Algoritma Genetika
- ✓ Hal-hal yang harus dilakukan dalam Algoritma Genetika
- ✓ Siklus Algoritma Genetika
- ✓ Hal Penting yang harus diperhatikan dalam pemakaian Algoritma Genetika
- ✓ Contoh Penggunaan Algoritma Genetika

TUJUAN BELAJAR:

Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan :

- ✓ Memahami Konsep Dasar Algoritma Genetika
- ✓ Memahami hal-hal yang harus dilakukan jika menggunakan Algoritma Genetika
- ✓ Mengetahui contoh penggunaan Algoritma Genetika
- ✓ Mampu menerapkan Algoritma Genetika dalam permasalahan yang lain

Algoritma Genetika sebagai cabang dari Algoritma Evolusi merupakan metode *adaptive* yang biasa digunakan untuk memecahkan suatu pencarian nilai dalam sebuah masalah optimasi. Algoritma ini didasarkan pada proses genetik yang ada dalam makhluk hidup; yaitu perkembangan generasi dalam sebuah populasi yang alami, secara lambat laun mengikuti prinsip seleksi alam atau "siapa yang kuat, dia yang bertahan (*survive*)". Dengan meniru teori evolusi ini, Algoritma Genetika dapat digunakan untuk mencari solusi permasalahan-permasalahan dalam dunia nyata.

Peletak prinsip dasar sekaligus pencipta Algoritma Genetika adalah John Holland. Algoritma Genetika menggunakan analogi secara langsung dari kebiasaan

yang alami yaitu seleksi alam. Algoritma ini bekerja dengan sebuah populasi yang terdiri dari individu-individu, yang masing-masing individu mempresentasikan sebuah solusi yang mungkin bagi persoalan yang ada. Dalam kaitan ini, individu dilambangkan dengan sebuah nilai *fitness* yang akan digunakan untuk mencari solusi terbaik dari persoalan yang ada.

Pertahanan yang tinggi dari individu memberikan kesempatan untuk melakukan reproduksi melalui perkawinan silang dengan individu yang lain dalam populasi tersebut. Individu baru yang dihasilkan dalam hal ini dinamakan keturunan, yang membawa beberapa sifat dari induknya. Sedangkan individu dalam populasi yang tidak terseleksi dalam reproduksi akan mati dengan sendirinya. Dengan jalan ini, beberapa generasi dengan karakteristik yang bagus akan bermunculan dalam populasi tersebut, untuk kemudian dicampur dan ditukar dengan karakter yang lain. Dengan mengawinkan semakin banyak individu, maka akan semakin banyak kemungkinan terbaik yang dapat diperoleh.

Sebelum Algoritma Genetika dapat dijalankan, maka sebuah kode yang sesuai (representatif) untuk persoalan harus dirancang. Untuk ini maka titik solusi dalam ruang permasalahan dikodekan dalam bentuk kromosom/string yang terdiri atas komponen genetik terkecil yaitu gen. Dengan teori evolusi dan teori genetika, di dalam penerapan Algoritma Genetika akan melibatkan beberapa operator, yaitu:

1. Operasi Evolusi yang melibatkan proses seleksi (*selection*) di dalamnya.
2. Operasi Genetika yang melibatkan operator pindah silang (*crossover*) dan mutasi (*mutation*).

Untuk memeriksa hasil optimasi, kita membutuhkan fungsi *fitness*, yang menandakan gambaran hasil (solusi) yang sudah dikodekan. Selama berjalan, induk harus digunakan untuk reproduksi, pindah silang dan mutasi untuk menciptakan keturunan. Jika Algoritma Genetika didesain secara baik, populasi akan mengalami konvergensi dan akan didapatkan sebuah solusi yang optimum.

7.1 HAL-HAL YANG HARUS DILAKUKAN DALAM ALGORITMA GENETIKA

Beberapa hal yang harus dilakukan dalam Algoritma Genetika adalah:

- ◆ **Mendefinisikan individu**, dimana individu menyatakan salah satu solusi (penyelesaian) yang mungkin dari permasalahan yang diangkat.
- ◆ **Mendefinisikan nilai fitness**, yang merupakan ukuran baik-tidaknya sebuah individu atau baik-tidaknya solusi yang didapatkan.
- ◆ Menentukan proses **pembangkitan populasi awal**. Hal ini biasanya dilakukan dengan menggunakan pembangkitan acak seperti *random-walk*.
- ◆ Menentukan proses **seleksi** yang akan digunakan.
- ◆ Menentukan proses **perkawinan silang (cross-over)** dan **mutasi gen** yang akan digunakan.

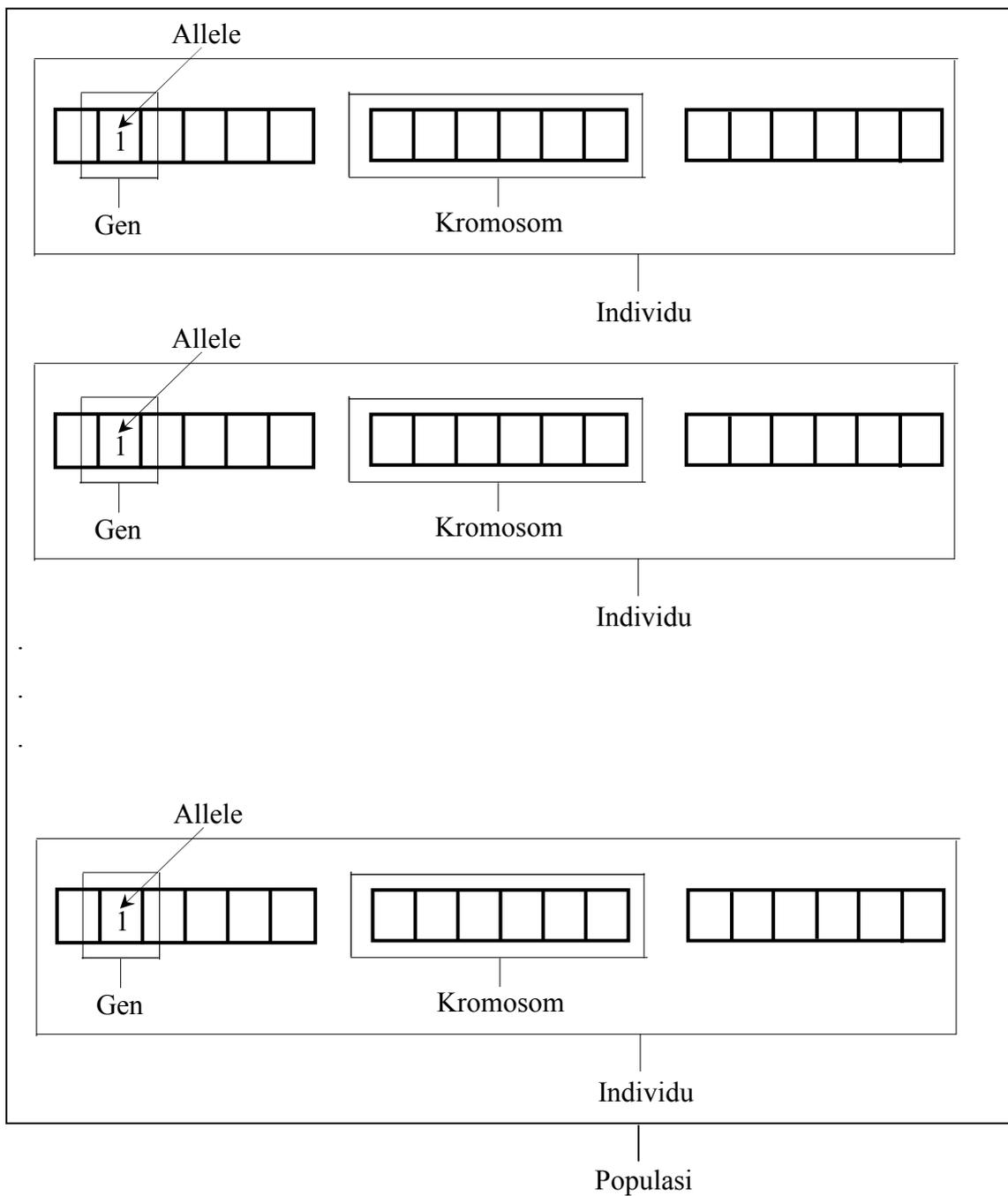
7.1.1 PENGERTIAN INDIVIDU

Individu menyatakan salah satu solusi yang mungkin. Individu bisa dikatakan sama dengan kromosom, yang merupakan kumpulan gen. Gen ini bisa biner, float, dan kombinatorial.

Beberapa definisi penting yang perlu diperhatikan di mendefinisikan individu untuk membangun penyelesaian permasalahan dengan algoritma genetika adalah sebagai berikut:

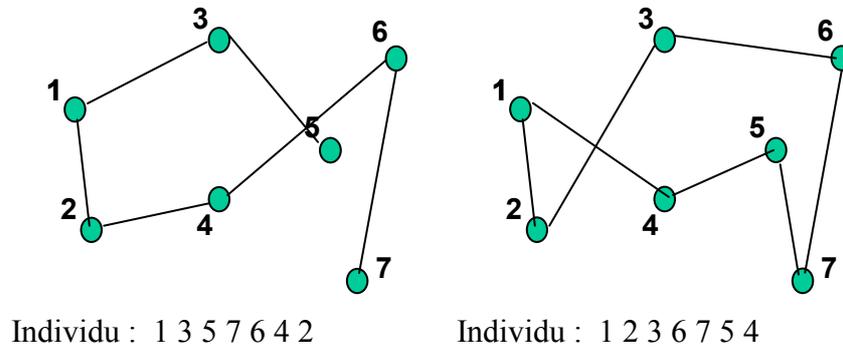
- ◆ **Genotype (Gen)**, sebuah nilai yang menyatakan satuan dasar yang membentuk suatu arti tertentu dalam satu kesatuan gen yang dinamakan kromosom. Dalam algoritma genetika, gen ini bisa berupa nilai biner, float, integer maupun karakter, atau kombinatorial.
- ◆ **Allele**, nilai dari gen.
- ◆ **Kromosom**, gabungan gen-gen yang membentuk nilai tertentu.
- ◆ **Individu**, menyatakan satu nilai atau keadaan yang menyatakan salah satu solusi yang mungkin dari permasalahan yang diangkat
- ◆ **Populasi**, merupakan sekumpulan individu yang akan diproses bersama dalam satu siklus proses evolusi.
- ◆ **Generasi**, menyatakan satu siklus proses evolusi atau satu iterasi di dalam algoritma genetika.

Pada gambar 7.1 diilustrasikan perbedaan istilah-istilah di atas.



Gambar 7.1 Ilustrasi Representasi Penyelesaian Permasalahan dalam Algoritma Genetika

Misalnya di dalam TSP individu menyatakan jalur yang ditempuh, dalam penentuan nilai maksimal dari $F(x,y)$ individu menyatakan nilai (x,y) . Pada gambar 7.2 diilustrasikan 2 kemungkinan jalur yang ditempuh dalam TSP dan bagaimana representasinya dalam individu.



Gambar 7.2 Kemungkinan jalur dalam TSP dan Representasi dalam individu

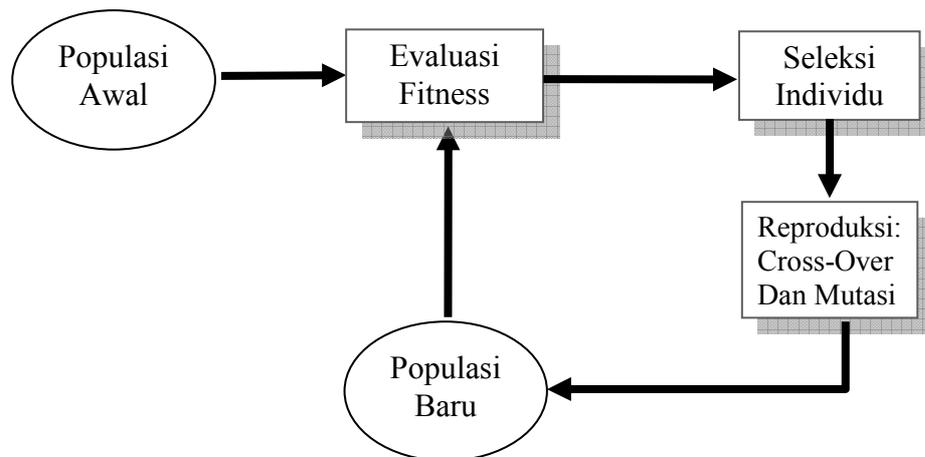
7.1.2 NILAI FITNESS

Nilai fitness adalah nilai yang menyatakan baik tidaknya suatu solusi (individu). Nilai fitness ini yang dijadikan acuan dalam mencapai nilai optimal dalam algoritma genetika. Algoritma genetika bertujuan mencari individu dengan nilai fitness yang paling tinggi.

Dalam TSP, karena TSP bertujuan meminimalkan jarak, maka nilai fitnessnya adalah inversi dari total jarak dari jalur yang didapatkan. Cara melakukan inversi bisa menggunakan rumus $1/x$ atau $100000-x$, dimana x adalah total jarak dari jalur yang didapatkan.

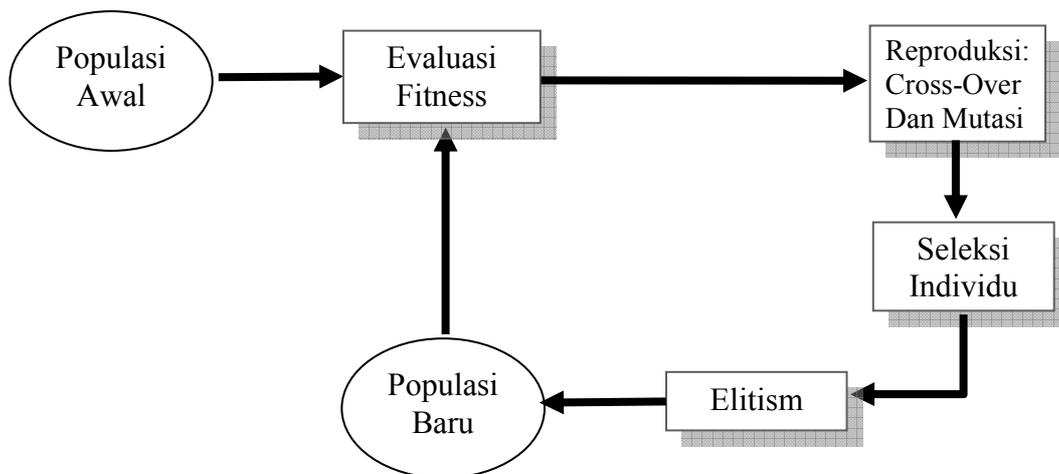
7.2 SIKLUS ALGORITMA GENETIKA

Siklus dari Algoritma Genetika pertama kali dikenalkan oleh David Goldberg, dimana gambaran siklus tsb. dapat dilihat pada gambar 7.3.



Gambar 7.3 Siklus Algoritma Genetika oleh David Goldberg

Siklus ini kemudian diperbaiki oleh beberapa ilmuwan yang mengembangkan algoritma genetika, yaitu Zbigniew Michalewicz dengan menambahkan operator elitism dan membalik proses seleksi setelah proses reproduksi.



Gambar 7.4 Siklus Algoritma Genetika yang diperbarui oleh Michalewicz

7.3 KOMPONEN-KOMPONEN UTAMA ALGORITMA GENETIKA

Terdapat 6 komponen utama dalam algoritma genetika, yaitu:

7.3.1 TEKNIK PENGKODEAN

Teknik pengkodean adalah bagaimana mengkodekan gen dari kromosom, dimana gen merupakan bagian dari kromosom. Satu gen biasanya akan mewakili satu variabel.

Gen dapat direpresentasikan dalam bentuk : bit, bilangan real, daftar aturan, elemen permutasi, elemen program atau representasi lainnya yang dapat diimplementasikan untuk operator genetika.

Dengan demikian kromosom dapat direpresentasikan dengan menggunakan:

- String bit : 10011 dst.
- Array bilangan real : 65.65, -67.98, 77.34 dst.
- Elemen permutasi : E2, E10, E5 dst
- Daftar aturan : R1, R2, R3 dst.
- Elemen program : pemrograman genetika
- Struktur lainnya.

7.3.2 MEMBANGKITKAN POPULASI AWAL

Membangkitkan populasi awal adalah proses membangkitkan sejumlah individu secara acak atau melalui prosedur tertentu. Ukuran untuk populasi tergantung pada masalah yang akan diselesaikan dan jenis operator genetika yang akan diimplementasikan. Setelah ukuran populasi ditentukan, kemudian dilakukan pembangkitan populasi awal. Syarat-syarat yang harus dipenuhi untuk menunjukkan suatu solusi harus benar-benar diperhatikan dalam pembangkitan setiap individunya.

Teknik dalam pembangkitan populasi awal ini ada beberapa cara, diantaranya adalah sebagai berikut:

1. Random Generator

Inti dari cara ini adalah melibatkan pembangkitan bilangan random untuk nilai setiap gen sesuai dengan representasi kromosom yang digunakan. Jika menggunakan representasi biner, salah satu contoh penggunaan random generator adalah penggunaan rumus berikut untuk pembangkitan populasi awal :

$$IPOP = \mathbf{round}\{\mathbf{random}(N_{ipop}, N_{bits})\}$$

dimana IPOP adalah gen yang nantinya berisi pembulatan dari bilangan random yang dibangkitkan sebanyak N_{ipop} (Jumlah populasi) \times N_{bits} (Jumlah Gen dalam tiap kromosom).

Contoh lain penggunaan random generator dalam representasi permutasi adalah pada saat dibangkitkan populasi awal untuk penyelesaian permasalahan Traveling Salesman Problem. Sebagai contoh, sebuah kromosom untuk 9 kota bisa direpresentasikan

[0.23 0.82 0.45 0.74 0.87 0.11 0.56 0.69 0.78]

di mana posisi i dalam list menunjukkan kota i . Nilai acak dalam posisi i menentukan urutan didatanginya kota i dalam lintasan TSP. Dengan kunci-kunci random di atas, kita dapat menentukan bahwa nilai 0.11 adalah yang paling kecil, sehingga kota ke-6 menempati urutan pertama, 0.23 adalah nilai terkecil kedua, sehingga kota ke-1 menempati urutan kedua dst. Sehingga dengan demikian, dari kunci-kunci random di atas kita dapat menentukan lintasan:

6 – 1 – 3 – 7 – 8 – 4 – 9 – 2 – 5

2. Pendekatan Tertentu (Memasukkan Nilai Tertentu ke dalam Gen)

Cara ini adalah dengan memasukkan nilai tertentu ke dalam gen dari populasi awal yang dibentuk.

3. Permutasi Gen

Salah satu cara permutasi gen dalam pembangkitan populasi awal adalah penggunaan permutasi Josephus dalam permasalahan kombinatorial seperti TSP. Misalkan ada kota dari 1 sampai 9. Permutasi dari lintasan dapat dilakukan dengan menentukan titik awal dan selang. Misalnya titik awal adalah 6 dan selang adalah 5. Maka lintasan berangkat dari kota 6, selang 5 dari kota 6 adalah kota 2 (dengan asumsi kota 1 sampai 9 membentuk *circular list*). Kota 2 dihapus dari list. Selang 5 kemudian adalah kota 7. Proses ini diulang hingga ada satu lintasan dalam list. Hasil dari permutasi ini adalah 2 – 7 – 3 – 8 – 4 – 9 – 5 – 1 – 6.

7.3.3 SELEKSI

Seleksi digunakan untuk memilih individu-individu mana saja yang akan dipilih untuk proses kawin silang dan mutasi. Seleksi digunakan untuk mendapatkan calon induk yang baik. “Induk yang baik akan menghasilkan keturunan yang baik”. Semakin tinggi nilai fitness suatu individu semakin besar kemungkinannya untuk terpilih.

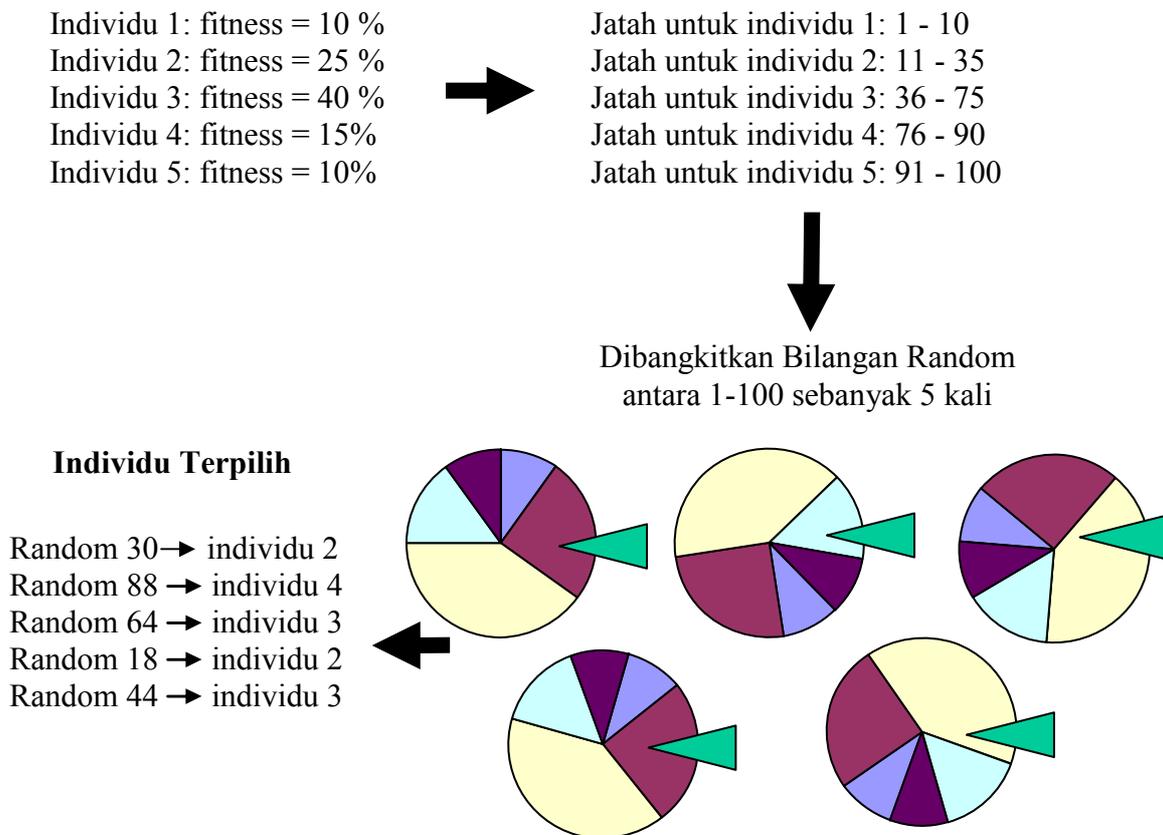
Langkah pertama yang dilakukan dalam seleksi ini adalah pencarian nilai fitness. Nilai fitness ini yang nantinya akan digunakan pada tahap-tahap seleksi berikutnya. Masing-masing individu dalam wadah seleksi akan menerima probabilitas reproduksi yang tergantung pada nilai obyektif dirinya sendiri terhadap nilai obyektif dari semua individu dalam wadah seleksi tersebut.

Terdapat beberapa metode seleksi, dalam buku ini hanya dibahas 2 metode yaitu mesin roulette, dan turnamen.

7.3.3.1 SELEKSI DENGAN MESIN ROULETTE

Metode seleksi dengan mesin roulette ini merupakan metode yang paling sederhana dan sering dikenal dengan nama stochastic sampling with replacement. Cara kerja metode ini adalah sebagai berikut:

1. Dihitung nilai fitness dari masing-masing individu (f_i , dimana i adalah individu ke-1 s/d ke- n)
2. Dihitung total fitness semua individu
3. Dihitung probabilitas masing-masing individu
4. Dari probabilitas tersebut, dihitung jatah masing-masing individu pada angka 1 sampai 100
5. Dibangkitkan bilangan random antara 1 sampai 100
6. Dari bilangan random yang dihasilkan, ditentukan individu mana yang terpilih dalam proses seleksi



Gambar 7.5 Ilustrasi Seleksi dengan MesirRoulette

7.3.3.2 SELEKSI DENGAN TURNAMEN

Pada metode seleksi dengan turnamen, ditetapkan suatu nilai tour untuk individu-individu yang dipilih secara random dari suatu populasi. Individu-individu yang terbaik dalam kelompok ini akan diseleksi sebagai induk. Parameter yang digunakan pada metode ini adalah ukuran tour yang bernilai antara 2 sampai N (jumlah individu dalam suatu populasi).

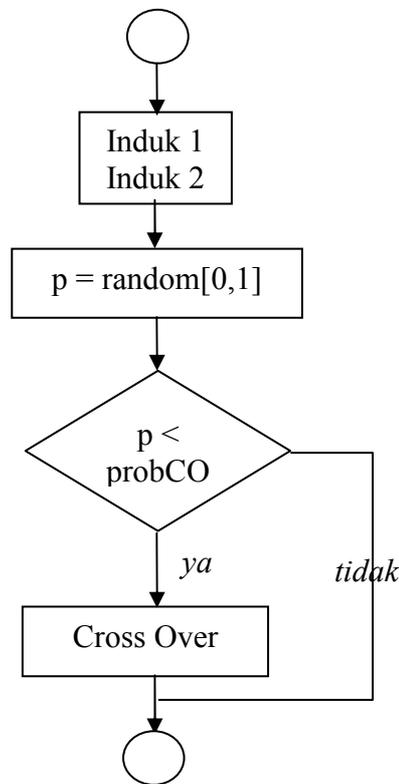
7.3.4 PINDAH SILANG (CROSSOVER)

Kawin silang (crossover) adalah operator dari algoritma genetika yang melibatkan dua induk untuk membentuk kromosom baru. Pindah silang menghasilkan titik baru dalam ruang pencarian yang siap untuk diuji. Operasi ini tidak selalu dilakukan pada semua individu yang ada. Individu dipilih secara acak untuk dilakukan

crossing dengan P_c antara 0,6 s/d 0,95. Jika pindah silang tidak dilakukan, maka nilai dari induk akan diturunkan kepada keturunan.

Prinsip dari pindah silang ini adalah melakukan operasi (pertukaran, aritmatika) pada gen-gen yang bersesuaian dari dua induk untuk menghasilkan individu baru. Proses crossover dilakukan pada setiap individu dengan probabilitas crossover yang ditentukan. Pada Gambar 7.6 diilustrasikan diagram alir penggunaan probabilitas crossover pada proses crossover.

Operator crossover ini bergantung pada representasi kromosom yang dilakukan. Berbagai model crossover sesuai dengan representasi kromosom akan diuraikan pada sub bab selanjutnya.

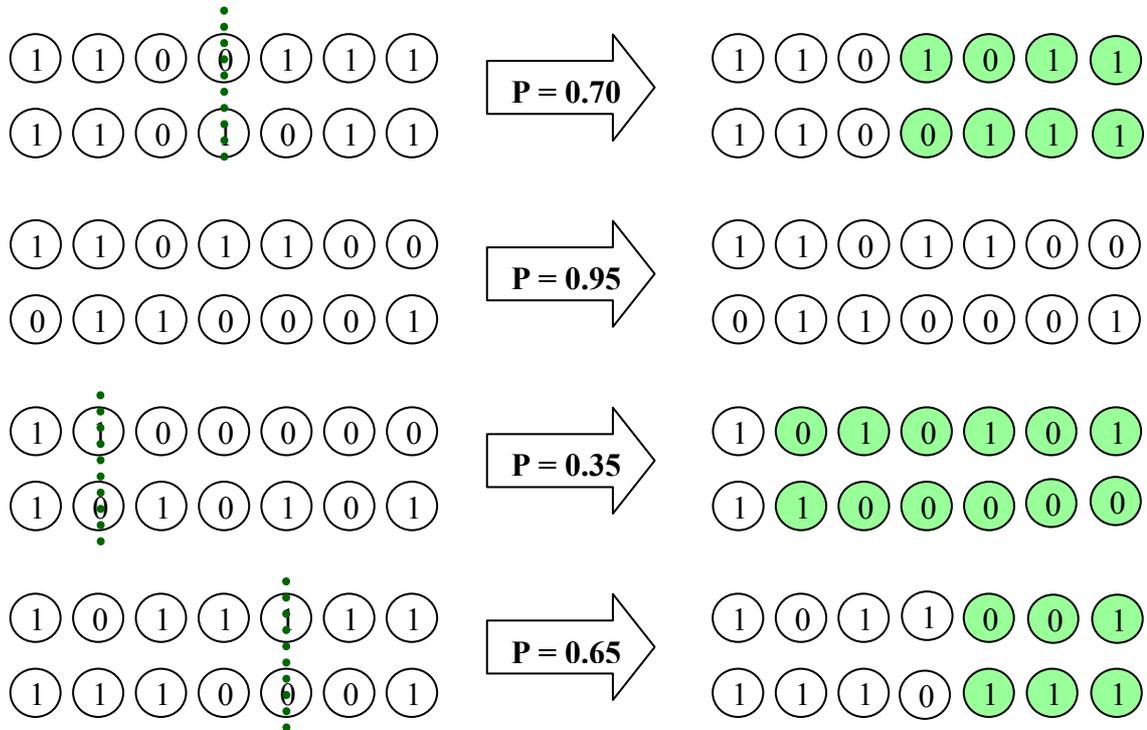


Gambar 7.6 Diagram Alir Proses Crossover

7.3.4.1 CROSSOVER SATU TITIK

Crossover satu titik dan banyak titik biasanya digunakan untuk representasi kromosom dalam biner. Pada crossover satu titik, posisi crossover k ($k=1,2,\dots,N-1$) dengan N =panjang kromosom diseleksi secara random. Variabel-variabel ditukar antar

kromosom pada titik tersebut untuk menghasilkan anak. Pada Gambar 7.7 diilustrasikan Crossover satu titik.



Ditentukan probabilitas Cross-Over = 0.9

Gambar 7.7 Ilustrasi Crossover Satu Titik

7.3.4.2 CROSSOVER BANYAK TITIK

Pada crossover banyak titik, m posisi penyilangan k_i ($k=1,2,\dots,N-1, i=1,2,\dots,m$) dengan N =panjang kromosom diseleksi secara random dan tidak diperbolehkan ada posisi yang sama, serta diurutkan naik. Variabel-variabel ditukar antar kromosom pada titik tersebut untuk menghasilkan anak. Pada Gambar 7.7 diilustrasikan Crossover Dua Titik dan pada Gambar 7.8 diilustrasikan Crossover lebih dari dua titik.

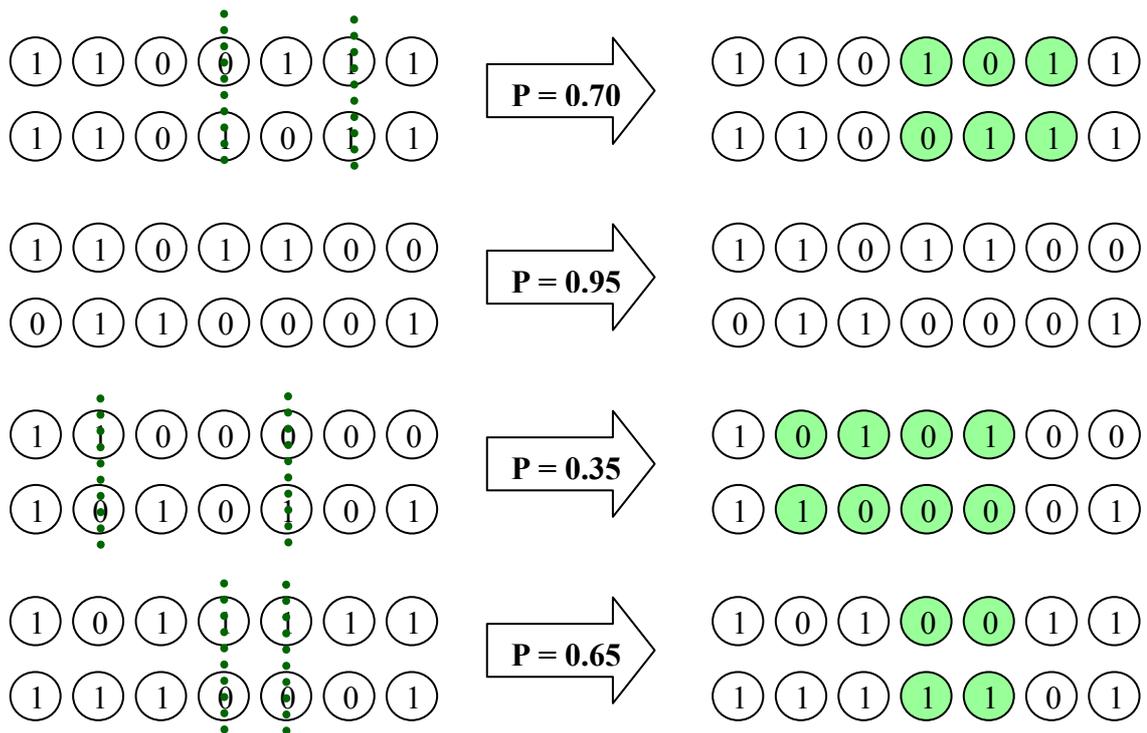
7.3.4.3 CROSSOVER ARITMATIKA

Crossover aritmatika digunakan untuk representasi kromosom berupa bilangan float (pecahan). Crossover ini dilakukan dengan menentukan nilai r sebagai bilangan random lebih dari 0 dan kuran dari 1. Selain itu juga ditentukan posisi dari gen yang dilakukan crossover menggunakan bilangan random. Pada Gambar 7.9 diilustrasikan

bagaimana crossover aritmatika bekerja. Nilai baru dari gen pada anak mengikuti rumus 7.1 dan rumus 7.2.

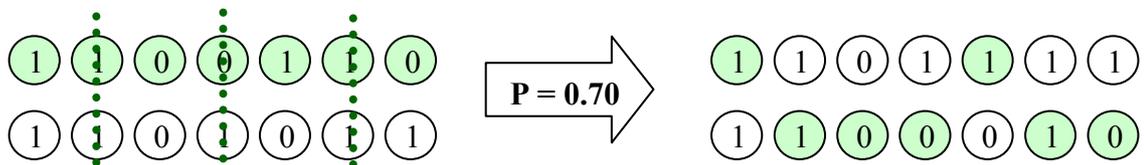
$$x_1'(k) = r \cdot x_1(k) + (1-r) \cdot x_2(k) \dots\dots\dots(7.1)$$

$$x_2'(k) = r \cdot x_2(k) + (1-r) \cdot x_1(k) \dots\dots\dots(7.2)$$



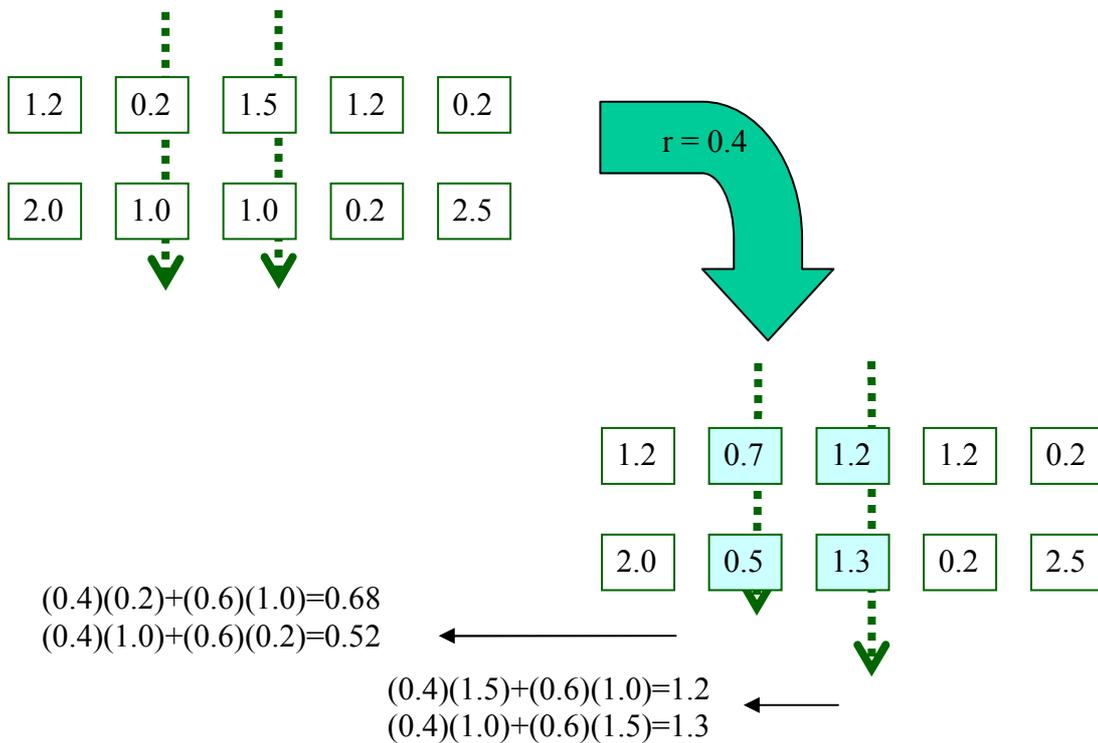
Ditentukan probabilitas Cross-Over = 0.9

Gambar 7.7 Ilustrasi Crossover Dua Titik



Ditentukan probabilitas Cross-Over = 0.9

Gambar 7.8 Ilustrasi Crossover Lebih Dua Titik



Gambar 7.9 Crossover Aritmatika

7.3.4.4 CROSSOVER UNTUK REPRESENTASI KROMOSOM PERMUTASI

Sejak pertengahan 80'an, beberapa metode operator pindah silang diciptakan untuk representasi permutasi, seperti *partial-mapped crossover*, *order crossover*, *cycle crossover*, *position-based crossover*, *order-based crossover*, *heuristic crossover*, dll. Beberapa metode pindah silang tersebut dijelaskan seperti di bawah ini.

Partial-Mapped Crossover (PMX). PMX diciptakan oleh Goldberg dan Lingle. PMX merupakan rumusan modifikasi dari pindah silang dua-poin. Hal yang penting dari PMX adalah pindah silang 2-poin ditambah dengan beberapa prosedur tambahan. PMX mempunyai langkah kerja sebagai berikut:

Prosedur PMX

Langkah 1: Tentukan dua posisi pada kromosom dengan aturan acak. Substring yang berada dalam dua posisi ini dinamakan daerah pemetaan.

Langkah 2: Tukar dua substring antar induk untuk menghasilkan *proto-child*.

Langkah 3: Tentukan hubungan pemetaan di antara dua daerah pemetaan.

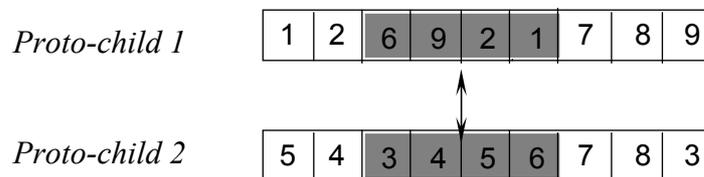
Langkah 4: Tentukan kromosom keturunan mengacu pada hubungan pemetaan.

Prosedur ini dapat dilihat ilustrasinya pada Gambar 7.10.

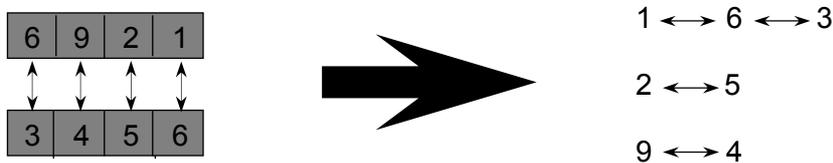
1. Pilih posisi untuk menentukan substring secara acak



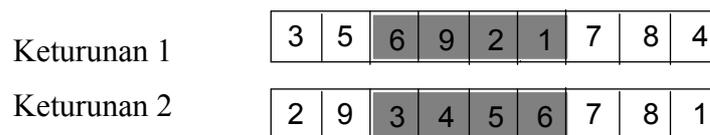
2. Tukar substring di antara induk



3. Menentukan hubungan mapping



4. Menentukan kromosom keturunan mengacu pada hubungan mapping



Gambar 7.10 Ilustrasi dari PMX operator

Order Crossover (OX). OX diciptakan oleh Davis. Metode ini merupakan variasi dari PMX dengan prosedur tambahan. OX bekerja sebagai berikut:

Prosedur OX

Langkah 1: Pilih substring dari sebuah induk secara acak.

Langkah 2: Bangkitkan sebuah *proto-child* dengan mengkosongkan tempat substring induk 2 pada induk 1.

Langkah 3: SHR allele dari substring pada tempat yang bersesuaian.

Langkah 4: Tukar substring antara 2 induk.

Gambar 7.11 adalah ilustrasi dari OX.

1. Memilih substring dari induk dengan cara acak

Induk 1	1	2	3	4	5	6	7	8	9
Induk 2	5	4	6	9	2	1	7	8	3

2. Produksi *proto-child* dengan mengosongkan tempat substring induk 2 pada induk 1

<i>Proto-child 1</i>			3	4	5		7	8	
<i>Proto-child 2</i>				9	2	1	7	8	

3. SHR substring pada tempat yang bersesuaian

<i>Proto-child 1</i>	7	8					3	4	5
<i>Proto-child 2</i>	7	8					9	2	1

4. Tukar posisi substring

Keturunan 1	7	8	6	9	2	1	3	4	5
Keturunan 2	7	8	3	4	5	6	9	2	1

Gambar 7.11 Ilustrasi OX operator

Cycle Crossover (CX). CX diciptakan oleh Oliver, Smith dan Holland. Metode ini mengkopi kota-kota dari satu induk dan memilih kota-kota yang lain dari induk yang lain, dengan mengingat dan pola cycle. Cara kerja CX adalah sbb:

Prosedur CX

Langkah 1: Temukan cycle yang didefinisikan dari relasi posisi kota-kota antara induk

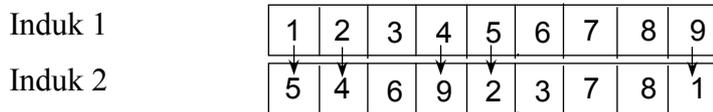
Langkah 2: Salin kota-kota dalam *cycle* pada *proto-child* dengan relasi posisi dari sebuah induk

Langkah 3: Tentukan kota-kota diingat yang berasal dari induk lain

Langkah 4: Isi keturunan dengan kota-kota yang diingat tadi

Gambar 7.12 adalah ilustrasi dari CX .

1. Tentukan pola *cycle* (asumsi : pola dimulai dari posisi 1)

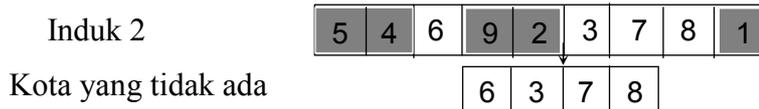


Pola *cycle* 1 → 5 → 2 → 4 → 9 → 1

2. Kopi kota-kota dalam *cycle* pada *proto-child*



3. Tentukan kota-kota yang diingat dari induk yang lain



3 Mengisikan kota yang diingat dalam keturunan



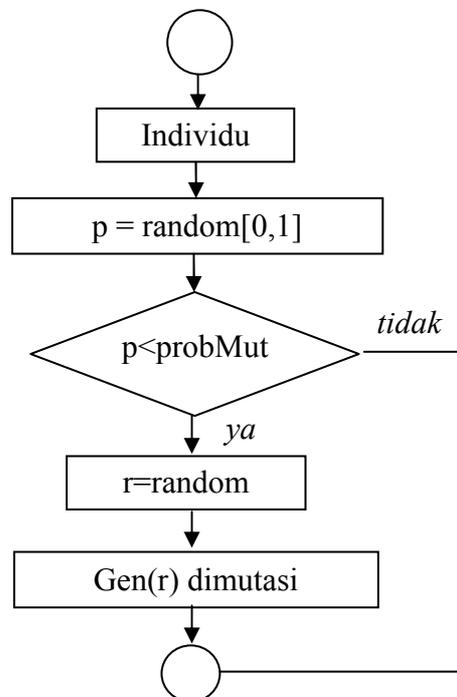
Dengan cara yang sama, diperoleh keturunan 2



Gambar 7.12 Ilustrasi CX operator

7.3.5 MUTASI

Operator berikutnya pada algoritma genetika adalah mutasi gen. Operator ini berperan untuk menggantikan gen yang hilang dari populasi akibat proses seleksi yang memungkinkan munculnya kembali gen yang tidak muncul pada inisialisasi populasi. Kromosom anak dimutasi dengan menambahkan nilai random yang sangat kecil (ukuran langkah mutasi), dengan probabilitas yang rendah. Peluang mutasi (p_m) didefinisikan sebagai persentase dari jumlah total gen pada populasi yang mengalami mutasi. Peluang mutasi mengendalikan banyaknya gen baru yang akan dimunculkan untuk dievaluasi. Jika peluang mutasi terlalu kecil, banyak gen yang mungkin berguna tidak pernah dievaluasi. Tetapi bila peluang mutasi ini terlalu besar, maka akan terlalu banyak gangguan acak, sehingga anak akan kehilangan kemiripan dari induknya, dan juga algoritma kehilangan kemampuan untuk belajar dari histori pencarian. Ada beberapa pendapat mengenai laju mutasi ini. Ada yang berpendapat bahwa laju mutasi sebesar $1/n$ akan memberikan hasil yang cukup baik. Ada juga yang beranggapan bahwa laju mutasi tidak tergantung pada ukuran populasinya. Kromosom hasil mutasi harus diperiksa, apakah masih berada pada domain solusi, dan bila perlu bisa dilakukan perbaikan.



Gambar 7.13 Diagram Alir Proses Mutasi

Pada Gambar 7.13 diilustrasikan diagram alir penggunaan probabilitas mutasi pada proses mutasi. Proses yang diilustrasikan tersebut adalah cara mudah untuk melakukan mutasi. Proses mutasi yang dilakukan tidak harus seperti pada proses tersebut. Proses yang lain bisa dengan melakukan mutasi pada gen sebanyak probabilitas mutasi * jumlah gen, dimana posisi gen yang akan dilakukan mutasi dipilih secara acak.

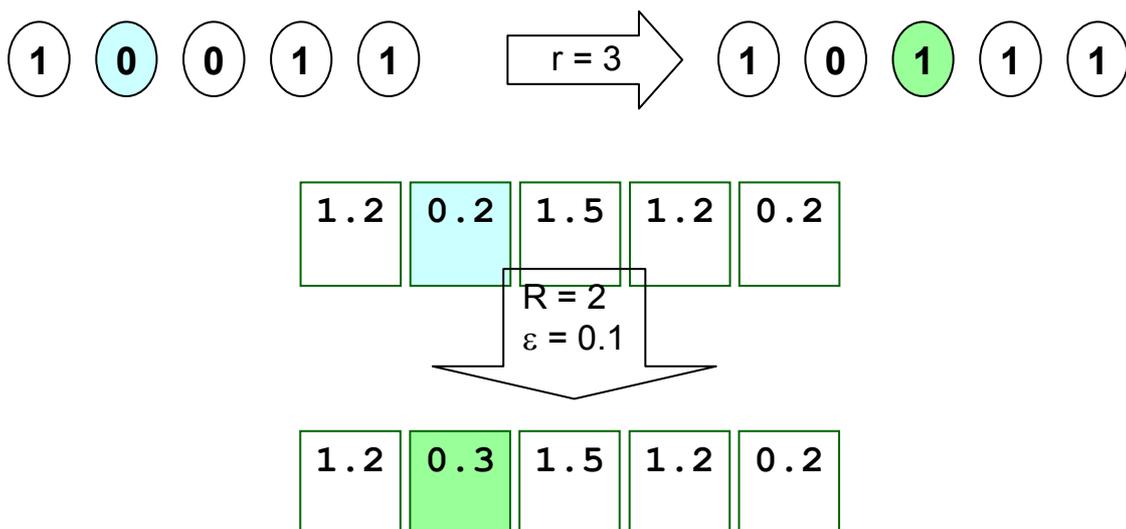
7.3.5.1 MUTASI BINER

Cara sederhana untuk mendapatkan mutasi biner adalah dengan mengganti satu atau beberapa nilai gen dari kromosom. Langkah-langkah mutasi ini adalah:

- Hitung jumlah gen pada populasi (panjang kromosom dikalikan dengan ukuran populasi).
- Pilih secara acak gen yang akan dimutasi.
- Tentukan kromosom dari gen yang terpilih untuk dimutasi.
- Ganti nilai gen (0 ke 1, atau 1 ke 0) dari kromosom yang akan dimutasi tersebut.

7.3.5.2 MUTASI BILANGAN REAL

7.3.5.3 MUTASI KROMOSOM PERMUTASI



Gambar 7.6 Proses & Hasil Mutasi

7.4 HAL PENTING YANG HARUS DIPERHATIKAN DALAM PEMAKAIAN ALGORITMA GENETIKA

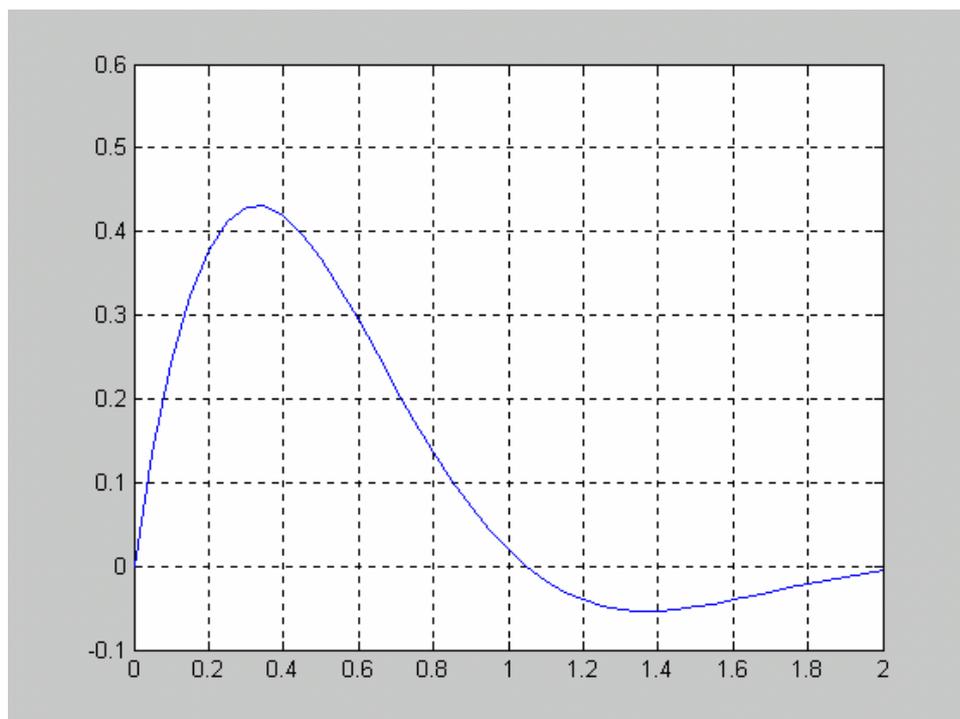
Beberapa hal yang perlu diperhatikan dalam pemakaian Algoritma Genetika adalah:

- Algoritma Genetika adalah algoritma yang dikembangkan dari proses pencarian solusi menggunakan pencarian acak, ini terlihat pada proses pembangkitan populasi awal yang menyatakan sekumpulan solusi yang dipilih secara acak.
- Berikutnya pencarian dilakukan berdasarkan proses-proses teori genetika yang memperhatikan pemikiran bagaimana memperoleh individu yang lebih baik, sehingga dalam proses evolusi dapat diharapkan diperoleh individu yang terbaik.

7.5 CONTOH PENGGUNAAN ALGORITMA GENETIKA

7.5.1 Algoritma Genetika Untuk Mencari Nilai Maksimal Fungsi $F(x)=e^{-2x}\cdot\sin(3x)$

Individu menyatakan nilai x , dalam mendefinisikan nilai x sebagai individu, dapat digunakan nilai biner atau nilai float. Pada algoritma genetika dasar digunakan nilai biner. Fungsi di atas bila digambarkan akan menjadi:



Dari gambar di atas terlihat bahwa penyelesaian berada pada nilai $0 < x < 1$. Jadi dengan menggunakan 8 bit biner didefinisikan :

00000000 berarti 0
11111111 berarti 1

7.5.1.1 DEFINISI INDIVIDU

Individu dinyatakan dalam 8 gen biner, dengan batas 0 sampai dengan 1, berarti 1 bit setara dengan 2-8.

Sebagai contoh:

$$10001001 = (128+8+1)/256 = 0.5352$$

$$00110100 = (4+16+32)/256 = 0.2031$$

$$01010010 = (2+16+64)/256 = 0.3203$$



7.5.1.2 FUNGSI FITNESS

Fungsi fitness adalah fungsi $f(x)$, karena yang dicari adalah nilai maksimum.

7.5.1.3 MEMBANGKITKAN POPULASI AWAL

Membangkitkan sejumlah individu, misalkan satu populasi terdiri dari 10 individu, maka dibangkitkan 10 individu dengan 8 gen biner yang dibangkitkan secara acak.

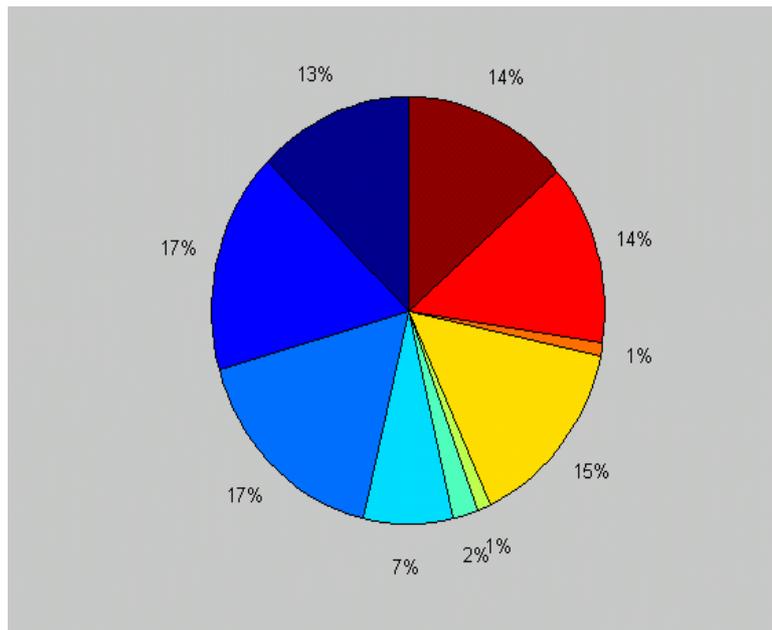
Individu	Fitness
10010000	-- 0.56250 -- 0.32244
01001110	-- 0.30469 -- 0.43060
01100110	-- 0.39844 -- 0.41933
10111101	-- 0.73828 -- 0.18266
11101000	-- 0.90625 -- 0.06699
11110010	-- 0.94531 -- 0.04543
00110011	-- 0.19922 -- 0.37778
11111100	-- 0.98438 -- 0.02616
10000111	-- 0.52734 -- 0.34828
10001011	-- 0.54297 -- 0.33702

→ Individu

7.5.1.4 SELEKSI

Seleksi adalah proses pemilihan calon induk, dalam proses seleksi ini terdapat beberapa metode yang bisa digunakan antara lain: Mesin Roulette (Roulette Wheel), Competition dan Tournament. Dalam contoh ini digunakan Mesin Roullete yang memang metode paling dasar dan model acaknya uniform.

Seleksi dilakukan dengan menggunakan prosentasi fitness setiap individu, dimana setiap individu mendapatkan luas bagian sesuai dengan prosentase nilai fitnessnya.



7.5.1.5 CROSSOVER

Cross-Over (Perkawinan Silang) merupakan proses mengkombinasikan dua individu untuk memperoleh individu-individu baru yang diharapkan mempunyai fitness lebih baik. Tidak semua pasangan induk mengalami proses cross-over, banyaknya pasangan induk yang mengalami cross-over ditentukan dengan nilai probabilitas cross-over.

0 0 1 1 1 0 0 1 -- 0.22266 ← induk 1	Fitness
1 0 0 1 1 0 1 0 -- 0.60156 ← induk 2	
0 0 1 1 1 0 1 1 -- 0.23050 ← anak 1	
1 0 0 1 1 0 0 0 -- 0.59382 ← anak 2	
	0.3968
	0.2921
	0.4022
	0.2982

7.5.1.6 MUTASI GEN

Mutasi gen adalah proses penggantian gen dengan nilai inversinya, gen 0 menjadi 1 dan gen 1 menjadi 0. Proses ini dilakukan secara acak pada posisi gen tertentu pada individu-individu yang terpilih untuk dimutasikan. Banyaknya individu yang mengalami mutasi ditentukan oleh besarnya probabilitas mutasi.

0 0 1 1 1 0 0 1 -- 0.22266 ← induk	Fitness
0 0 1 1 1 1 0 1 -- 0.22266 ← induk	
	0.3968
	0.4070

7.5.1.7 CONTOH HASIL ALGORITMA GENETIKA

Contoh hasil Algoritma Genetika untuk permasalahan ini adalah:

Generasi ke 1 :

```
10100111 -- 0.65234 -- 0.25127
01000110 -- 0.27344 -- 0.42328
01001110 -- 0.04297 -- 0.43060
01110110 -- 0.46094 -- 0.39076
10111001 -- 0.72266 -- 0.19488
10001111 -- 0.55859 -- 0.32540
10001000 -- 0.53125 -- 0.34550
10010011 -- 0.57422 -- 0.31348
00111011 -- 0.23047 -- 0.40214
10000011 -- 0.51172 -- 0.35913
```

Generasi ke 2 :

```
10000000 -- 0.50000 -- 0.36696
10001010 -- 0.53906 -- 0.33987
01001110 -- 0.04297 -- 0.43060
10010111 -- 0.58984 -- 0.30132
10000011 -- 0.51172 -- 0.35913
10000011 -- 0.51172 -- 0.35913
01001110 -- 0.30469 -- 0.43060
10001000 -- 0.53125 -- 0.34550
10111101 -- 0.73828 -- 0.18266
01000010 -- 0.25781 -- 0.41715
```

Generasi ke 3 :

```
01001110 -- 0.30469 -- 0.43060
01001110 -- 0.30469 -- 0.43060
10000011 -- 0.51172 -- 0.35913
10001000 -- 0.53125 -- 0.34550
11001110 -- 0.80469 -- 0.13301
01001110 -- 0.04297 -- 0.43060
10001010 -- 0.53906 -- 0.33987
10000011 -- 0.51172 -- 0.35913
01001110 -- 0.30469 -- 0.43060
10000011 -- 0.51172 -- 0.35913
```

Generasi ke 4 :

```
00001110 -- 0.05469 -- 0.14641
11001000 -- 0.78125 -- 0.15005
01001110 -- 0.30469 -- 0.43060
10000011 -- 0.51172 -- 0.35913
01001110 -- 0.30469 -- 0.43060
01001110 -- 0.30469 -- 0.43060
01000011 -- 0.26172 -- 0.41885
10001110 -- 0.55469 -- 0.32833
00001110 -- 0.05469 -- 0.14641
01001110 -- 0.78906 -- 0.43060
```

7.6 RINGKASAN

- ✓ Algoritma Genetika adalah Sistem Kecerdasan Buatan yang meniru sel pada tubuh manusia.
- ✓ Algoritma Genetika dapat digunakan untuk proses pencarian dan proses optimasi.
- ✓ Untuk menggunakan Algoritma Genetika ada beberapa langkah yang harus dilakukan : **Mendefinisikan individu**, **Mendefinisikan nilai fitness**, **pembangkitan populasi awal**, Menentukan proses **seleksi** yang akan digunakan, Menentukan proses **perkawinan silang (cross-over)** dan **mutasi gen** yang akan digunakan.
- ✓ Beberapa hal yang perlu diperhatikan dalam pemakaian Algoritma Genetika adalah:

Algoritma Genetika adalah algoritma yang dikembangkan dari proses pencarian solusi menggunakan pencarian acak, ini terlihat pada proses pembangkitan populasi awal yang menyatakan sekumpulan solusi yang dipilih secara acak.

- ✓ Berikutnya pencarian dilakukan berdasarkan proses-proses teori genetika yang memperhatikan pemikiran bagaimana memperoleh individu yang lebih baik, sehingga dalam proses evolusi dapat diharapkan diperoleh individu yang terbaik.

7.7 LATIHAN

- 1) Sebutkan beberapa langkah yang harus dilakukan jika kita menggunakan Algoritma Genetika!
- 2) Apa definisi dari Fungsi Fitness ?
- 3) Bagaimanakah fungsi fitness dalam permasalahan TSP?
- 4) Definisikan individu, fungsi fitness, model kawin silang, mutasi jika kita menyelesaikan persoalan rumus :

$$f(x, y) = 2 + e^{-(x^2+y^2)} \{ \sin(4x) + \cos(8y) \}$$