BAB VII
ALGORITMA GENETIKA

Kompetensi :
1. Mahasiswa memahami konsep Algoritma Genetika

Sub Kompetensi :
1. Dapat mengerti dasar metode Algoritma Genetika
2. Dapat memahami tahapan operator dalam Algoritma Genetika

VII.1. Pengertian Dasar Metode Algoritma Genetika

Algoritma genetika nerupakan algoritma pencarian yang berdasarkan pada seleksi alam dan genetika alam. Algoritma ini berguna untuk masalah yang memerlukan pencarian yang efektif dan efisien, dan dapat digunakan secara meluas untuk aplikasi bisnis, pengetahuan, dan dalam ruang lingkup teknik. Algoritma genetika ini dapat digunakan untuk mendapatkan solusi yang tepat untuk masalah satu atau banyak variabel.

Algoritma genetika berbeda dengan teknik pencarian yang lain, karena pada algoritma genetika ini langkah pertama dimulai dengan membangkitkan secara random solusi-solusi yang sering dikenal dengan *initial population*. Setiap individu di dalam populasi dinamakan *chromosome*, dimana setiap *chromosome* itu mewakili sebuah solusi untuk masalah yang akan dihadapi. Sebuah *chromosome* biasanya simbolnya string hal ini diperuntukkan bagi bilangan biner dan untuk *floating poin*t yang dipakai adalah bilangan real. Untuk masalah tiga variabel maka *chromosome* akan tersusun atas tiga gen demikian pula kalau permasalahannya melibatkan lima variabel, maka didalam satu *chromosome* juga akan terdapat lima gen. *Chromosome* terbentuk setiap generasi dan kemudian dievaluasi menggunakan beberapa ukuran *fitness* **.** Untuk generasi yang baru, *chromosome* baru terbentuk oleh proses yang dinamakan proses seleksi . Setelah proses seleksi itu berlangsung *chromosome* yang baru terbentuk itu akan mengalami proses reproduksi dimana didalam proses reproduksi ini *chromosome* tadi akan daproses dalam dua tahap yaitu *crossover* dan mutasi. Kadua tahap proses itu akan membuat *offspring.* Untuk proses *crossover* **,** *offspring* yang terbentuk merupakan penggabungan dari *chromosome* yang sebelumnya, sedangkan untuk mutasi *offspring* yang terbentuk merupakan hasil perubahan mutasi dari gen atau mutasi pada bit.

Generasi baru terbentuk oleh seleksi menurut nilai fitness dari keseluruhan *chromosome* , beberapa *parent* dan *offspring* dipilih agar menjaga ukuran populasi tetap. *Chromosome* yang memiliki nilai fitness yang besar memiliki peluang yang lebih besar untuk terpilih. Setelah beberapa generasi, Algoritma Genetika ini akan mengumpulkan *chromosome* terbaik, yang membantu untuk mewakili solusi yang optimal untuk masalah itu.

Ada 2 mekanisme yang menghubungkan Algoritma Genetika dengan masalah yang ingin diselesaikan yaitu cara pengkodean /*encoding* penyelesaian untuk masalah pada *chromosome* dan fungsi evaluasi yang mengembalikan ukuran harga dari setiap *chromosome* dalam konteks dari masalah itu. Teknik untuk mengkodekan penyelesaian bermacam-macam dari masalah ke masalah dan dari Algoritma Genetika ke Algoritma Genetika .

Populasi disusun dari bermacam-macam individu, setiap individu berisi phenotype (parameter), genotype (*chromosome* buatan atau *bit string*), dan *fitness* (fungsi obyektif). Fungsi evaluasi adalah penghubung antara Algoritma Genetika dan masalah yang akan diselesaikan. Fungsi evaluasi mengambil sebuah *chromosome* sebagai input dan mengembalikan nomor atau daftar dari nomor yang merupakan ukuran dari *chromosome* pada masalah yang akan diselesaikan.

Fungsi evaluasi mempunyai aturan yang sama dalam Algoritma Genetika yang berkecimpung dalam evolusi alam .

Inisialisasi dilakukan secara random atau acak. Rekombinasi termasuk di dalamnya *crossover* / kawin silang dan mutasi untuk munghasilkan *offspring* / keturunan. Kenyataannya ada dua operasi pada Algoritma Genetika

- Operasi genetika yaitu *crossover* dan mutasi
- Operasi evolusi yaitu seleksi

Operasi genetika meniru proses turun menurun dari gen untuk membentuk *offspring* pada setiap generasi. Operasi evolusi meniru proses dari *Darwinian evolution* untuk membentuk populasi dari generasi ke generasi .

Untuk memudahkan pengertian *chromosome*, Algoritma Genetika akan diterapkan pada contoh persamaan berikut ini .

$$\text{Max } f(x,y) = 21{,}5 + x\sin(4\pi x) + y\sin(20\pi y) \qquad (1)$$

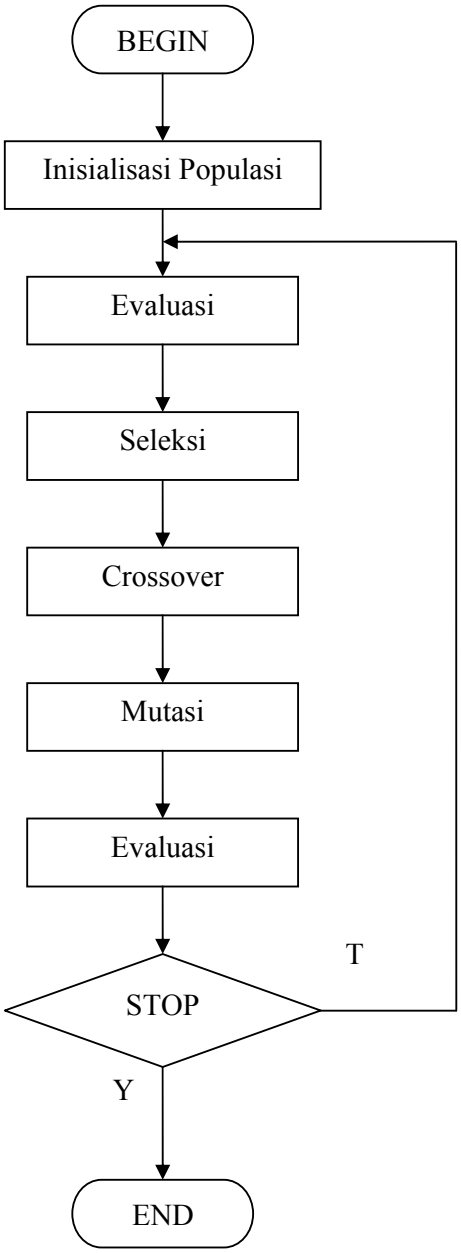
Dimana batasannya :

$$-3 \le x \le 12{,}1 \qquad (2)$$

$$4{,}1 \le y \le 5{,}8$$

Pada persamaan 2.35 ini maka *chromosome* terbentuk dari dua buah gen yaitu x dan y sehingga bentuk dari c*hromosome* adalah (x,y). Sedangkan untuk inisialisasi populasi

pertama kali apabila diambil 10 *chromosome* dalam satu populasi, maka nilai dari x akan diambil secara acak dari antara -3 dan 12,1 sebanyak 10 kali demikian juga untuk nilai y akan diambil secara acak dari antara 4,1 dan 5,8 sebanyak 10 kali. Nilai dari x pertama kali akan membentuk *chromosome* pertama, demikian untuk nilai-nilai yang lain sampai nilai x kesepuluh dan nilai yang kesepuluh akan membentuk *chromosome* kesepuluh dengan susunan *chromosome* seperti di atas yaitu (x,y). Blok diagram untuk proses algoritma genetika diberikan pada gambar 3.1 berikut ini.



Gambar 1. Blok Diagram Algoritma Genetika

VII.2. Operator Algoritma Genetika

Sama seperti suatu penyelesaian umum lainnya, Algoritma Genetika mempunyai operator-operator yang secara berurutan dipakai untuk menyelesaikan suatu permasalahan. Operator-operator tersebut dibahas berikut ini.

A. Seleksi

Tiga dasar pokok yang tergabung di dalam fase seleksi dalam metode Algoritma Genetika yaitu:

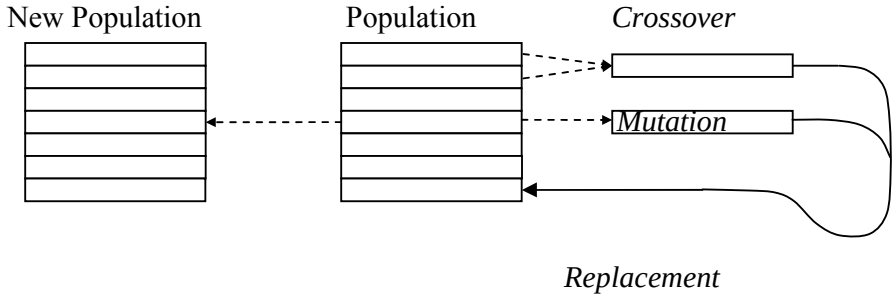
- *Sampling space*
- *Sampling mechanism*
- *Selection probability*

Sampling Space

Prosedur seleksi yang menghasilkan populasi baru untuk generasi selanjutnya berdasarkan pada baik semua *parent* atau *offspring* atau bagian dari mereka. Inilah yang memimpin masalah dari *sampling space*. Sebuah *sampling space* dikarakteristikkan oleh dua faktor yaitu *size* (ukuran) dan *ingredient* (bahan) yang menunjuk pada *parent* dan *offspring*. *Pop_size* menunjuk pada ukuran dari populasi dan *off_size* menunjuk pada ukuran dari *offspring* yang dihasilkan pada setiap generasi. *Regular sampling space* memiliki ukuran dari *pop_size* dan berisi semua *offspring* dengan sebagian dari *parent*. *Enlarge sampling place* memiliki okoran dari *pop_size* + *off_size* dan berisi semua *parent* dan *offspring*.

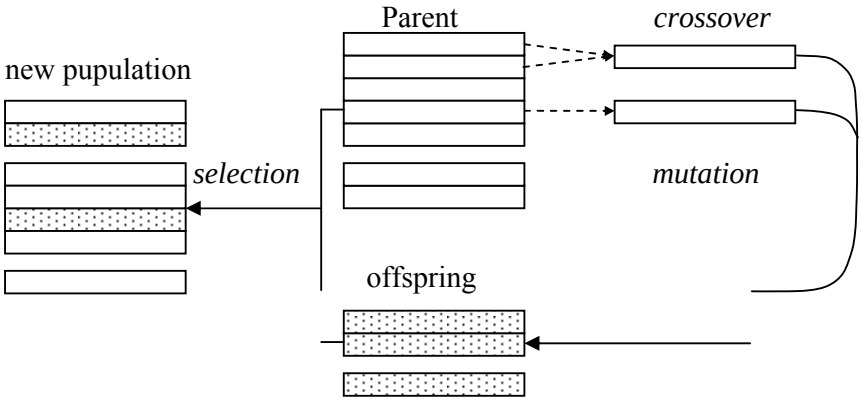
- *Regular sampling space*: pada waktu terjadi *crossover* dan *mutasi* itulah maka akan dilahirkan *offspring*. *Parent* digantikan oleh *offspring* atau keturunannya segera setelah kelahirannya. Pergantian secara langsung ini dinamakan *generational repacement*. Karena operasi genetika buta dalam alam, *offspring* mungkin lebih buruk dibandingkan dengan *parent*. Strategi pergantian setiap *parent* dengan *offspring* secara langsung akan menyebabkan beberapa *chromosome* yang memiliki peluang yang besar akan hilang dari proses evolusi . Ada beberapa ide untuk mengatasi masalah ini . Holland mengusulkan ketika setiap *offspring* lahir, *offspring* itu akan menggantikan secara random *chromosome* yang dipilih dari populasi sekarang. Dalam model *crowding*, ketika *offspring* lahir, satu *parent* yang dipilih akan mati. *Parent* yang mati itu dipilih dengan ciri seperti *parent* yang mirip *offspring* yang baru, kemiripan ini diperoleh dengan menggunakan kesamaan bit per bit untuk menghitung ukuran kemiripannya. Pada *reproductive plan*, seleksi menunjuk pada *parent* yang terpilih untuk rekombinasi, populasi yang baru terbentuk oleh pergantian *parent*

dengan *offspring* yang terbentuk. Michalewics memberikan deskripsi pada Algoritma Genetika sederhana dimana *offspring* dalam setiap generasi akan menggantikan *parent* untuk generasi tersebut segera setelah kelahirannya, generasi yang baru dibentuk oleh *roulette wheel selection.* Gambar 3.2 berikut ini adalah merupakan seleksi yang dilakukan pada regular *sampling space.*



Gambar 2. Seleksi berdasarkan pada *Regular Sampling Space*

- *Enlarge sampling space*: ket*ika seleksi dilakukan pada enlarge sampling space, baik parent dan offspring* mempunyai kesempatan untuk berkompetisi dengan tujuan untuk bertahan. Untuk sebuah kasus khusus adalah seleksi ($\mu + \lambda$). Stategi ini digunakan dalam stategi evolusi. Dengan stategi ini , μ *parent* dan λ *offspring* berkomperisi untuk kelangsungan hidupnya dan μ terbaik keluar dari *offspring* dan *old parent* dipilih sebagai *parent* pada generasi selanjutnya. Kasus lain dari strategi evolusi adalah seleksi (μ,λ), dimana memilih μ *offspring* terbaik sebagai *parent* pada generasi selanjutnya. Bila dilihat dari gambar 3.3 berikut ini maka *parent* setelah mengalami *crossover* dan mutasi akan menghasilkan *offspring* yang akan ditambahkan ke *parent.* Untuk menjaga agar populasi itu konstan, maka pada waktu dilakukan seleksi dibatasi hanya untuk menghasilkan beberapa *chromosome* tergantung pada ukuran populasi.



Gambar 3. Seleksi Dilakukan Pada *Enlarge Sampling Space*

Sampling Mechanism

Sampling mechanism memperhatikan masalah bagaimana memilih *chromosome* dari *sampling space*. Tiga dasar pendekatan yang telah digunakan pada *sampling chromosome*:

- *Stochastic sampling*
- *Deterministic sampling*
- *Mixed sampling*

a. *Stochastic sampling*

Ciri-ciri umum pada metode ini yaitu fase seleksi menentukan jumlah dari penggandaan yang akan diterima setiap *chromosome* berdasarkan pada probabilitas *survival.* Fase seleksi disusun dari dua bagian:

- Menentukan nilai *chromosome* yang diharapkan
- Mengkonversi nilai yang diharapkan pada nomor dari *offspring*

Nilai yang diharapkan dari *chromosome* adalah nomor real yang mengindikasikan nomor rata-rata dari *offspring* yang seharusnya diterima sebuah *chromosome.* Prosedur *sampling* digunakan untuk mengubah nilai real yang diharapkan pada nomor dari *offspring.* Ide dasar dari *Roulette Wheel Selection* untuk menentukan probabilitas seleksi (probabilitas *survival*) untuk setiap *chromosome* sebanding pada nilai *fitness.* Untuk *chromosome k* dengan *fitness* f_k , probabilitas seleksi P_k dihitung dari:

$$p_k = \frac{f_k}{\sum_{j=1}^{pop\_size} f_j} \qquad (3)$$

Kita dapat membuat sebuah *wheel* menurut probabilitas ini. Proses seleksi berdasarkan pada putaran *roulette wheel pop_size,* setiap waktu kita memilih *chromosome* tunggal untuk populasi yang baru. Baker memperkenalkan *Stochastic Universal Sampling,* yang menggunakan putaran roda tunggal (*single - wheel spin*). Roda disusun seperti *roulette wheel,* nilai yang diharapkan e_k untuk *chromosome* k dihitung sebagai:

$$e_k = pop\_size \; x \; p_k \qquad (4)$$

Prosedur dari *Stochastic Universal Sampling:*

```
begin
sum:=0;
ptr:=rand();
    for k:= 1 to pop_size do
        sum:= sum + ek ;
```

```
        while (sum > ptr) do
        select chromosome k ;
        ptr:= ptr + 1;
      end
   end
end
```

Dimana rand() adalah sebuah jumlah real random yang secara *uniform* didistribusikan diantara *range* 0 sampai 1. Tujuan dasarnya adalah menjaga jumlah pengkopian yang diharapkan dari setiap *chromosome* dalam generasi berikutnya. Perhatian yang menarik adalah *prohibition of duplicate chromosome* (larangan penggandaan *chromosome*) dalam populasi.

Dua alasan menggunakan strategi ini adalah :

1. Mencegah *chromosome* super dari pendominisasian populasi oleh penggunaan banyak pengkopian dalam populasi.
2. Menjaga perbedaan-perbedaan dari populasi sehingga kelompok generasi yang konstan dapat berisi banyak informasi untuk pencarian genetika.

Masalah yang terkait adalah ketika *chromosome* duplikat dibuang, ukuran dari populasi terbentuk dari sisa *parent* dan *offspring* mungkin lebih rendah daripada ukuran *pop_size* sebelumnya. Pada kasus ini, prosedur inisialisasi biasanya digunakan untuk mengisi *space* kosong dari kelompok populasi.

b. *Deterministic sampling*

Pendekatan ini biasanya memilih *pop_size chromosome* terbaik dari *sampling space.* Baik seleksi $(\mu+\lambda)$ dan seleksi (μ,λ) termasuk dalam metode ini. Dari hasil yang ingin dicapai adalah melarang *chromosome- chromosome* duplikat masuk populasi pada waktu seleksi. Seleksi *Elitist* menjamin bahwa *chromosome* terbaik masuk generasi yang baru kalau tidak terpilih melalui proses seleksi yang lain. Modifikasi pada metode ini adalah dengan mengganti sejumlah n *chromosome* tua yang jelek dengan *offspring* (dengan jumlah n *offspring).*

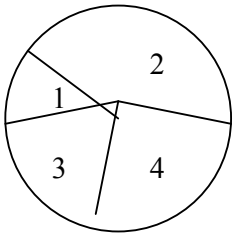
c. *Mixed sampling*

Pendekatan ini berisi baik random dan ciri-ciri *deterministic* secara bersama. Contoh dari *moxed sampling* adalah *Tournament Selection*. Metode ini secara random memilih sekumpulan *chromosome* dan mengambil satu terbaik dari kumpulan untuk reproduksi. Jumlah *chromosome* dalam kumpulan dinamakan *Tournament Size.* Ukuran *tournament* umumnya adalah dua, ini yang dinamakan *Binary Tournament* dan *Stochastic Tournament*

Selection oleh Wetzel, pada metode ini probabilitas seleksi dihitung secara normal dan berturut-turut pasangan dari *chromosome-chromosome* digambarkan menggunakan *Roulette Wheel Selection.* Setelah menggambarkan sepasang, *chromosome* dengan *fitness* yang tinggi dimasukkan dalam populasi yang baru. Proses ini berlangsung secara kontinu sampai populasi penuh.

Selection Probability

Bagian ini membicarakan bagaimana probabilitas seleksi untuk setiap *chromosome*. Dalam prosedur seleksi perbandingan, probabilitas seleksi dari sebuah *chromosome* sebanding dengan *fitness value* yang dimilikinya. Contoh dalam generasi awal, dimana ada sebuah kecenderungan untuk sebuah super *chromosome* kecil akan mendominasi proses seleksi. Dalam generasi selanjutnya ketika populasi secara besar berkumpul, kompetisi diantara *chromosome* kurang kuat dan pencarian random tingkah laku akan muncul. *Scaling* dan *ranking mechanism* diharapkan dapat mengurangi masalah ini. Metode *scaling* membuat peta mentah nilai fungsi obyektif menjadi beberapa nilai real positif, dan probabilitas *survival* untuk setiap *chromosome* menurut nilai itu. Matode *ranking* menganggap nilai aktual fungsi obyektif dan menggunakan *ranking* dari *chromosome* sebagai pengganti untuk menentukan probabilitas *survival*. Proses seleksi adalah proses pemilihan *chromosome*. Setelah *chromosome* itu terpilih maka *chromosome* itu langsung akan menjadi *chromosome* awal pada generasi yang baru. Dengan kata lain setelah melewati proses seleksi itu maka akan berganti ke generasi baru. Proses seleksi yang umum digunakan adalah dengan menggunakan *Roulette Wheel Selection*. seleksi *Roulette Wheel* diasumsikan dengan model *pie-shaped.* Pada gambar 3.4, nomor 1, 2, 3, dan 4 menyatakan nomor *chromosome*.



Gambar 4 Model *Roulette Wheel*

Pada gambar 4 ini empat daerah luasan yang ada mewakili empat *chromosome* dimana setiap *chromosome* memiliki luas daerah yang berbeda. Perbedaan itu muncul sebagai akibat dari perbedaan *fitness. Roulette wheel parent* dapat disusun seperti langkah-langkah berikut ini:

- Menghitung nilai *fitness* eval (v_k) untuk setiap *chromosome* v_k.

$$\text{eval}(V_k) = f(x) \qquad k = 1,2,3,\ldots..,\text{pop\_size} \qquad (5)$$

- Menghitung total *fitness* untuk populasi:

$$F = \sum_{k=1}^{\text{pop\_size}} \text{eval}(V_k) \qquad (6)$$

- Menghitung probabilitas seleksi p_k untuk setiap *chromosome* v_k:

$$P_k = \frac{\text{eval}(V_k)}{F} \qquad k = 1,2,3,\ldots.,\text{pop\_size} \qquad (7)$$

- Menghitung probabilitas akumulatif q_k untuk setiap *chromosome* v_k :

$$q_k = \sum_{f=1}^{k} P_f, \qquad k = 1,2,3,\ldots..,\text{pop\_size} \qquad (8)$$

Pada bagian ini prosedur seleksinya dimulai dari langkah:

- Membangkitkan nomor secara random r dari 0 sampai 1.
- Bila $q_1 \geq r$, maka pilih *chromosome* pertama v_1, kalau tidak pilih *chromosome* ke k atau v_k dimana (2 <= k <= pop_size) seperti q_{k-1} ($r \leq q_k$)

Beberapa pemakai Algoritma Genetika juga banyak yang menggunakan metode *Elitist* yaitu pemilihan *chromosome* yang memiliki *fitness* yang paling baik di dalam satu populasi. Pada pemilihan *chromosome* dalam suatu populasi yang terdiri dari 10 *chromosome,* pemilhan *chromosome* itu tidak mengurangi jumlah *chromosome* dalam satu populasi, melainkan metode *elitist* ini berfungsi mirip dengan penggandaan dan kemudian menyimpan hasil penggandaan itu. *Chromosome* yang terpilih dalam metode *elitist* ini tidak melewati urutan proses seperti seleksi, *crossover*, dan mutasi, tetapi *chromosome* yang terpilih akan menggantikan secara langsung *chromosome* pada generasi selanjutnya, yang memiliki nilai fitness paling kecil.

B. *Crossover*

Di alam, *crossover* terjadi ketika *parent* bertukar bagian dari gen pembentuknya. Dalam Algoritma Genetika, *crossover* menggabungkan materi genetika dari kedua *chromosome parent* menjadi dua anak. *Crossover* yang dipakai untuk bilangan real adalah *Arithmatic Crossover.*

Prosedur untuk memilih *parent* mana yang akan mengalami proses *crossover* :

- Tentukan probabilitas *crossover*.
- Bangkitkan bilangan random 0 sampai 1 sebanyak i (jumlah *chromosome* dalam satu populasi).
- Bandingkan bilangan random itu dengan probabilitas *crossover* (Pc).

- *Parent* terpilih bila bilangan r yang ke-i kurang atau sama dengan probabilitas *crossover* (Pc).
- Bila *parent* yang terpilih jumlahnya hanya satu maka proses ini diulang sampai jumlah *parent* lebih dari satu.

Aritmatic Chrossover

Proses selanjutnya setelah melakukan pemilihan *Chromosome* adalah melakukan *Aritmatic Crossover* yang dikerjakan untuk kondisi bilangan real pada kedua *parent* tersebut.

- Bangkitkan bilangan random antara 0 dan 1
- *Offspring*1 = (*parent*1 *x* random) + (*parent2 x* (1-random)).
- *Offspring*2 = (*parent*1 *x* (1-random)) + (*parent2 x* random)).

One-Point Crossover

Sedangkan untuk bilangan biner, pada proses *crossover* ada perbedaan. Untuk bilangan biner prosedur pemilihan *chromosome* sama dengan proses pemilihan *chromosome* seperti pada bilangan real. Untuk biner prosesnya adalah:

- Total bit merupakan banyaknya bit dalam satu populasi.
- Bangkitkan bilangan random antara 1 sampai total bit-1. Ini merupakan titik *crossover*, misalnya ditemukan nilai randomnya m.
- Pertukarkan bit yang ke (m +1) sampai total bit dari *parent* dengan bit yang ke (m+1) sampai ke total bit dari *parent 2.*

C.Mutasi

Proses mutasi adalah proses yang bertujuan untuk mengubah salah satu atau lebih bagian dari *chromosome.* Untuk bilangan *floating* ini memakai *Nonuniform Mutation* atau yang dikenal sebagai *Dynamic Mutation.* Mutasi ini didesain untuk dapat mentuning dengan baik dengan tujuan mencapai tingkat ketelitian yang tinggi. Prosedur untuk menentukan gen mana yang akan dimutasi adalah :

1. Tentukan probabilitas mutasi.
2. Tentukan banyaknya random yang diperoleh dari banyaknya jumlah *chromosome* dalam satu populasi *x* banyaknya gen dalam satu *chromosome* (total_random).
3. Bangkitkan bilangan random antara 0 dan 1 sebanyak total_random.
4. Bandingkan hasil random yang didapat sebanyak total_random dengan probabilitas mutasi.

5. Bila kurang dari probabilitas mutasi (Pm) maka gen tersebut yang akan dipilih untuk dimutasi.
6. Gen yang terpilih kemudian dihitung sehingga dapat diketahui gen tersebut berada pada *chromosome* nomor berapa dan pada gen yang nomor berapa.

Nonuniform Mutation

Proses mutasinya diberikan contoh *parent x,* bila elemen x_k dari *parent x* yang terpilih untuk dimutasi, hasil *offspring* adalah $x' = (x'_1,... \; x'_k ,,,, \; x'_n)$, dimana x'_k secara random terpilih dari dua pilihan yaitu :

$$x'_k = x_k + \Delta\left(t, x_k^U - x_k\right) \tag{3.9}$$

atau

$$x'_k = x_k - \Delta\left(t, x_k - x_k^L\right) \tag{3.10}$$

Fungsi $\Delta(t, y)$ diberikan

$$\Delta(t, y) = y \cdot r\left(1 - \frac{t}{T}\right)^b \tag{3.11}$$

dimana: r adalah bilangan random antara 0 dan 1

T adalah jumlah maksimal generasi

B adalah parameter yang menentukan derajat dari *nonuniformity*

t adalah nomor generasi

Binary Mutation

Mutasi ini memiliki prosedur yang mirip dengan yang digunakan oleh *nonuniform mutation.* Prosedur untuk pemilihan *chromosome*nya yaitu:

a. Tentukan probabilitas mutasi.
b. Tentukan banyaknya random yang diperoleh dari banyaknya jumlah *chromosome* dalam satu populasi x banyaknya jumlah bit dalam satu *chromosome* (total_random).
c. Bangkitkan bilangan random antara 0 dan 1 sebanyak total_random.
d. Bandingkan hasil random yang sebanyak total_random itu dengan probabilitas mutasi.
e. Bila nilai random itu kurang dari probabilitas mutasi (Pm) maka pilih bit itu untuk dimutasi.
f. Hitung bit yang terpilih itu berada pada *chromosome* ke berapa dan pada posisi bit yang keberapa.

Untuk proses mutasinya, setelah diketahui bit yang akan dimutasi itu berada pada *chromosome* ke berapa dan pada bit yang keberapa maka bila bit itu bernilai '0' maka berubah ke '1', sebaliknya bila '0' maka berubah ke '1'.

Free Direction Mutation

Mutasi dapat juga menggunakan cara sederhana seperti yang dipakai dalam tugas akhir ini. Kromosom yang ada dimutasi dengan "arah bebas". Maksudnya dapat bernilai positif maupun negatif . sebagai contoh , *parent* $v = (x_1, x_2,\ldots,x_n)$, arah mutasi dibangkitkan secara random d, sehingga *offspring* v' yang terbentuk ditentukan oleh rumus sebagai berikut : $v = v + M \cdot d$

Apabila nilai *offspring* yang terbentuk tidak berada pada *range* yang dikehendaki, set M Secara random sebagai bilangan real dalam (0, M) sedemikian sampai $v + M \cdot d$ berada pada *range* yang dikehendaki.

Urutan dari proses Algoritma Genetika adalah :

a. Inisialisasi populasi. Inisialisasi ini dilakukan secara random dan hanya satu kali saja sewaktu *start* pertama kali Algoritma Genetika . Inisialisasi ini menghasilkan populasi awal dengan jumlah *chromosome* yang sesuai dengan yang kita harapkan.
b. Evaluasi. Ini adalah proses menghitung nilai fitness dari masing-masing *chromosome* yang ada .
c. Seleksi. Melalui proses ini maka lahirlah generasi baru dimana *chromosome* diperoleh dari *chromosome* sebelumnya .
d. Crossover. Proses ini akan menghasilkan *offspring* yang berbeda dengan *psrent*/orang tuanya. Untuk proses ini dilakukan dengan mengambil *parent* secara berpasangan yaitu dua-dua dan akan menghasilkan *offspring* dengan jumlah yang sama dengan parent .
e. Mutasi. Pada proses ini juga akan dihasilkan *offspring* dimana jumlahnya tergantung pada banyaknya bilangan random yang kurang dari probabilitas mutasinya (pm).
f. Evaluasi. Proses ini dilakukan kembali untuk menghitung *fitness* dari masing-masing *chromosome*.
g. Lakukan pengulangan kembali ke langkah b, bila kriteria berhenti sudah dicapai maka Algoritma Genetika ini sudah selesai.

DAFTAR PUSTAKA

Duane Hanselman dan Bruce Littlefield. 2000. *MATLAB Bahasa Komputasi Teknis*. Yogya : ANDI

MATLAB & SIMULINK student version. 2004. *Learning MATLAB 7*. The MathWorks,Inc.

Suyanto. 2005. *Algoritma Genetika dalam Matlab.* Yogyakarta : Andi.

_____ . *Genetic Algorithm And Direct Search Toolbox*. The MathWorks,Inc.