BAB II
LANDASAN TEORI

2.1 Jadwal Mata Pelajaran

Menurut Jong Jek Siang (2000:22) jadwal merupakan sesuatu yang menjelaskan dimana dan kapan orang – orang dan sumber daya berada pada suatu waktu, Sedangkan penjadwalan merupakan proses cara pembuatan menjadwalkan atau memasukkan dalam jadwal.

Menurut Ana (2006:6) Mata pelajaran adalah suatu pelajaran dimana pelajaran tersebut selalu mempelajari hal-hal dalam kehidupan sehari-hari sehingga dengan memahami pelajaran tersebut kita lebih berhati-hati dalam menjalani kehidupan ini dan tidak menganggap enteng suatu masalah yang kecil padahal masalah yang kecil tersebut dapat mengakibatkan hal-hal yang besar atau suatu cabang pengetahuan yang diajarkan atau diteliti di tingkat perguruan tinggi.

Kebanyakan orang terbiasa dengan jadwal pelajaran yang disajikan sebagai tabel hari dalam seminggu dan jangka waktu. Dapat dilihat bahwa setiap hari dibagi ke dalam jangka waktu. Setiap jangka waktu memiliki daftar mata pelajaran yang sedang diajarkan, oleh siapa dan di mana.

2.2 Algoritma Genetika

Algoritma genetika adalah algoritma pencarian yang di dasarkan pada mekanisme seleksi alamiah dan genetika alamiah. Pada awalnya Algoritma genetika digunakan sebagai algoritma pencarian parameter-parameter optimal. Tetapi, dalam perkembanganya Algoritma genetika bisa diaplikasikan untuk berbagai masalah seperti *learning*, peramalan, pemrograman otomatis dan sebagainya.

Dalam proses seleksi alamiah individu secara terus-menerus mengalami perubahan gen untuk menyesuaikan dengan lingkungan hidupnya. individu-individu yang berniali *fitness* tinggi yang mampu bertahan sedangkan individu bernilai *fitness* rendah akan mati.

Algoritma genetika pertama kali dikembangkan oleh John Holland dari Univeritas Michigan (1975).

Algoritma genetika dibagi atas beberapa bagian diantaranya adalah sebagai berikut (Fadlisyah, Arnawan, Faisal ,2009:3) :

1. Gen

   Gen merupakan nilai yang menyatakan satuan dasar yang membentuk suatu arti tertentu dalam satu kesatuan gen yang dinamakan kromosom.

2. Kromosom / individu

   Kromosom merupakan gabungan dari gen gen yang membentuk nilai tertentu dan menyatakan solusi yang mungkin dari suatu permasalahan.

3. Populasi

   Populasi merupakan sekumpulan individu yang akan diproses bersama dalam satu satuan siklus evolusi.

4. *Fitness*

   Fitness menyatakan seberapa baik nilai dari suatu individu atau solusi yang didapatkan.

5. Seleksi

   Seleksi merupakan proses untuk mendapatkan calon induk yang baik.

6. *Crossover*

   Crossover merupakan proses pertukaran atau kawin silang gen gen dari dua induk tertentu.

7. Mutasi

   Mutasi merupakan proses pergantian salah satu gen yang terpilih dengan nilai tertentu.

8. Generasi

   Generasi merupakan urutan iterasi dimana beberapa kromosom bergabung.

9. *Offspring*

   Offspring merupakan kromosom baru yang dihasilkan setelah melewati suatu generasi.

2.2.1 Struktur Umum Algoritma Genetika

Beberapa hal yang harus dilakukan dalam Algoritma Genetika adalah Romauli Statina Sihombing (2014):

1. Mendefenisikan individu, dimana individu menyatakan salah satu solusi yang mungkin dari permasalahan yang diangkat.
2. Mendefenisikan nilai *fitness*, yang merupakan ukuran baik-tidaknya sebuah individu atau baik-tidaknya solusi yang didapatkan.
3. Menentukan proses pembangkitan populasi awal.
4. Menentukan proses seleksi yang akan digunakan.
5. Menentukan proses perkawinan silang (*crossover*) dan mutasi gen yang akan digunakan.

Variabel dan parameter yang digunakan pada algoritma genetika adalah Romauli Statina Sihombing (2014):

1. Fungsi *fitness* (fungsi tujuan) yang dimiliki oleh masing-masing individu untuk menentukan tingkat kesesuaian individu tersebut dengan kriteria yang ingin dicapai.
2. Populasi jumlah individu yang dilibatkan pada setiap generasi.
3. Probabilitas terjadinya persilangan (*Crossover*) pada suatu generasi.
4. Probabilitas terjadinya mutasi pada setiap individu.
5. Jumlah generasi yang akan dibentuk yang menentukan lama penerapan Algoritma Genetika.

2.2.2 Komponen Utama Algoritma Genetika

Ada 7 komponen utama dalam Algoritma Genetika:

1. Teknik penyandian

Teknik penyandian disini meliputi penyandian gen dari kromosom. Gen merupakan bagian dari kromosom. Satu gen biasanya akan mewakili satu variable. Gen dapat direpresentasikan dalam bentuk : *string bit*, pohon, *array*, bilanagn real, daftar aturan, elemen program, atau representasi lainnya yang dapat di implementasikan untuk operator genetika

Contoh dari *representasi* kromosom antara lain sebagai berikut :

1) String bit : 10011, 11101, dst
2) Bilangan Real : 65.65, 562.88, dst
3) Elemen Permutasi : E2, E10, dst
4) Daftar Aturan : R1, R2, R3, dst
5) Elemen Program : pemrograman genetika, dst
6) Struktur lainnya

2. Prosedur inisialisasi

Ukuran populasi tergantung pada permasalahan yang akan dipecahkan dan jenis operator genetika yang akan diimplementasikan. Setelah ukuran populasi telah ditentukan, kemudian harus dilakukan inisialisasi terhadap kromosom yang terdapat pada populasi tersebut. Inisialisasi kromosom dapat dilakukan secara acak, namun demikian harus tetap memperhatikan domain solusi dan kendala permasalahan yang ada.

3. Nilai *Fitness*

Suatu individu dievaluasi berdasarkan suatu fungsi tertentu sebagai ukuran performansinya. Didalam evolusi alam, individu yang bernilai *fitness* tinggi yang akan bertahan hidup. Sedangkan individu yang bernilai *fitness* rendah akan mati. Pada masalah optimasi, solusi yang akan dicari adalah memaksimumkan fungsi h (dikenal sebagai masalah maksimasi) sehingga nilai *fitness* yang digunakan adalah nilai dari fungsi h tersebut, yakni

f = h ……..…(2.1)

(di mana f adalah nilai *fitness*). Tetapi jika masalahnya adalah meminimalkan fungsi h (masalah minimasi), maka fungsi h tidak bisa digunakan secara langsung. Hal ini disebabkan adanya aturan bahwa individu yang memiliki nilai *fitness* tinggi lebih mampu bertahan hidup pada generasi berikutnya. Oleh karena itu nilai *fitness* yang bisa digunakan adalah

f = 1/h ……..…(2.2)

yang artinya semakin kecil nilai h, semakin besar nilai f. Tetapi hal ini akan menjadi masalah jika h bisa bernilai 0, yang mengakibatkan f bisa bernilai tak

hingga. Untuk mengatasinya, h perlu ditambah sebuah bilangan yang dianggap kecil [0-1] sehingga nilai *fitness*-nya menjadi :

$$f = \frac{1}{(h+a)} \qquad (2.3)$$

(suyanto, 2011, Artifical intelligence)

4. Seleksi

Memiliki tujuan untuk memberikan kesempatan reproduksi yang lebih besar bagi anggota populasi yang paling fit. Seleksi akan menentukan individu-individu mana saja yang akan dipilih untuk dilakukan rekombinasi dan bagaimana *Offspring* terbentuk dari individu-individu terpilih tersebut. Langkah pertama yaitu pencarian nilai *fitness*. Langkah kedua adalah nilai *fitness* yang diperolah digunakan pada tahap-tahap seleksi selanjutnya.

Ada beberapa metode seleksi dari induk (Sri kusumadewi, 2003) yaitu :

1. ***Rank-based fitness***

   Populasi diurutkan menurut nilai objektifnya. Nilai *fitness* dari tiap-tiap individu hanya tergantung pada posisi individu tersebut dalam urutan, dan tidak dipengaruhi oleh nilai objektifnya

2. **Seleksi Roda Roulette (*Roulette wheel selection*)**

   Metode ini Individu -individu dipetakan dalam suatu segmen garis secara berurutan sehingga tiap-tiap segmen individu memiliki ukuran yang sama dengan ukuran *fitness*nya. Sebuah bilangan random dibangkitkan dan individu yang memiliki segmen dalam kawasan segmen dalam kawasan bilangan random tersebut akan terseleksi. Proses ini berulang hingga didapatkan sejumlah individu yang diharapkan.

3. ***Stochastic universal sampling***

   Memiliki nilai bias nol dan penyebaran yang minimum. Individu-individu dipetakan dalam suatu segmen garis secara berurut sedemikian hingga tiap-tiap segmen individu memiliki ukuran yang sama dengan ukuran *fitness*nya seperti halnya pada seleksi roda roulette. Kemudian diberikan sejumlah pointer sebanyak individu yang ingin diseleksi pada garis tersebut. Andaikan N adalah jumlah individu yang akan diseleksi, maka

jarak antar pointer adalah1/N, dan posisi pointer pertama diberikan secara acak pada range [1, 1/N].

4. Seleksi *Lokal (Local selection)*

Pada seleksi lokal setiap individu yang berada di dalam konstraint tertentu disebut dengan nama lingkungan lokal. Interaksi antar individu hanya dilakukan dalam wilayah tersebut. Lingkungan tersebut ditetapkan sebagai struktur dimana populasi tersebut terdistribusi. Lingkungan tersebut juga dipandang sebagai sekelompok pasangan-pasangan yang potensial.

5. Seleksi dengan Pemotongan (*Truncation selection*)

Seleksi ini biasanya digunakan oleh populasi yang jumlahnya sangat besar. Pada metode inin individu-individu diurutkan berdasarkan nilai *fitness*-nya. Hanya individu yang terbaik saja yang akan diseleksi sebagai induk. Parameter yang digunakan adalah suatu nilai ambang trunc yang mengindikasikan ukuran populasi yang akan diseleksi sebagai induk yang berkisar antara 50% - 10%. Individu-individu yang ada dibawah nilai ambang tidak akan menghasilkan keturunan.

6. Seleksi dengan Turnamen (*Tournament selection*)

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 adalah ukuran tour yang bernilai antara 2 sampai N (jumlah individu dalam populasi).

5. Pindah silang

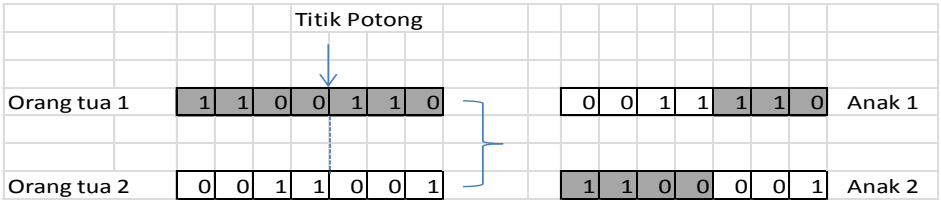
Pada proses pindah silang terjadi kombinasi pewarisan gen-gen dari induknya, gen-gen dari kedua induk dapat bercampur sehingga dihasilkan susunan kromosom yang baru. Dari proses tersebut akan dihasilkan variasi genetik.

Dengan suatu skema tertentu, dua individu dipilih sebagai orang tua. Setelah didapatkan dua individu orang tua, selanjutnya ditentukan titik pindah silang secara acak. Kemudian beberapa bagian dari dua kromosom ditukar pada titik pindah silang yang dipilih. Titik pindah silang adalah titik terjadinya pertukaran gen antara dua individu orang tua. Pertukaran tersebut akan menghasilkan dua

individu anak. Bagaimanapyun, operasi pindah silang tidak selamanya berhasil. Peluang keberhasilan operasi pindah silang inyatakan dengan probabilitas pindah silang atau *pc*. Terdapat tiga skema pindah silang yang biasa digunakan (suyanto, 2011, Artifical intelligence) yaitu:

1. **Pindah silang satu titik (*single –point Crossover*)**

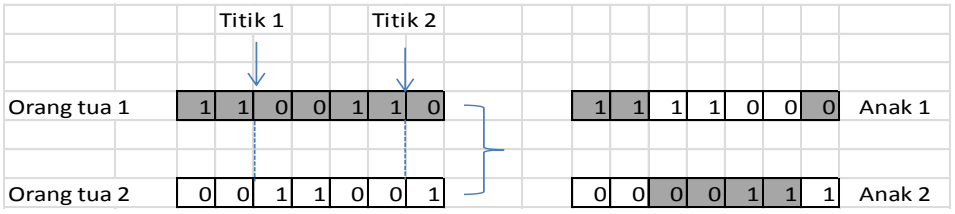
   Pindah silang ini merupakan skema pindah silang yang paling sederhana. Titik pindah silang hanya satu dengan posisi yang dibangkitkan secara acak.



Gambar 2.1 Contoh pindah silang satu titik

2. **Pindah silang banyak titik (*multi –point Crossover*)**

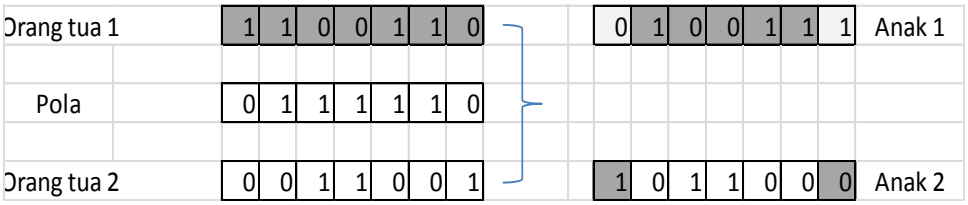
   Pada suatu masalah tertentu dimana suatu individu terdiri dari sangat banyak gen (misalkan 10000 gen), mungkin kita memerlukan lebih dari satu titik pindah silang . banyaknya titik pindah silang ini akan mempengaruhi pola pertukaran gen-gen antar individu orang tua.



Gambar 2.2 Contoh pindah silang banyak titik (lebih dari satu). Dalam contoh ini terdapat dua titik pindah silang yang dibangkitkan secara acak

3. **Pindah silang pola seragam (*uniform Crossover*)**

   Dengan operasi pindah silang pola seragam maka komposisi gen-gen tertentu pada suatu individu dapat dipertahankan. hal ini akan memudahkan proses pencarian solusi.



Gambar 2.3 Contoh pindah silang pola seragam. Dalam contoh ini pindah silang dilakukan berdasarkan suatu pola tertentu. Pindah silang dilakukan jika pola bernilai 0.

6. Mutasi

Mutasi diperlukan untuk mengembalikan informasi bit yang hilang akibat *crossover*. Mutasi ditetapkan dengan probabilitas yang sangat kecil. Jika mutasi dilakukan terlalu sering , maka akan menghasilkan individu yang lemah karena konfigurasi gen pada individu yang unggul akan rusak. Berdasarkan bagian yang termutasi, proses mutasi dapat dibedakan atas tiga bagian (suyanto, 2011, Artifical intelligence) yaitu:

1. Mutasi pada tingkat kromosom semua gen dalam kromosom berubah.

Gambar 2.4 Contoh mutasi tingkat kromosom. Semua gen dalam kromosom berubah . pada contoh ini, gen yang terjadi bernilai 0 berubah menjadi 1 dan gen yang tadinya bernilai 1 berubah menjadi 0

2. Mutasi pada tingkat gen: semua bit dalam satu gen akan berubah. Misal gen 2 yang mengalami mutasi.

Gambar 2.5 Contoh mutasi tingkat gen. semua bit dalam suatu gen berubah

3. Mutasi pada tingkat hanya satu bit yang berubah

Gambar 2.6 Contoh mutasi tingkat bit. Hanya satu bit yang berubah

7. Parameter Kontrol

Parameter kontrol dalam algoritma genetika diperlukan untuk mengendalikan operator-operator seleksi. Pemilihan parameter genetika menentukan penampilan kinerja algoritma genetika dalam memecahkan masalah. Ada dua parameter dasar dari algoritma genetika, yaitu probabilitas *crossover* (Pc) dan probabilitas mutasi (Pm)

1. Probabilitas *crossover*

   Probabilitas *crossover* menyatakan seberapa sering proses *crossover* akan terjadi antara dua kromosom orang tua. Jika tidak terjadi *crossover*, satu orang tua dipilih secara random dengan probabilitas yang sama dan diduplikasi menjadi anak. Jika terjadi *crossover*, keturunan dibuat dari bagian-bagian kromosom orang tua. Jika probabilitas *crossover* 100% maka keseluruhan keturunan baru dibuat dengan *crossover* . Namun sebaliknya jika probabilitas *crossover* 0% maka seluruh generasi baru dibuat dari salinan kromosom-kromosom populasi lama yang belum tentu menghasilkan populasi yang sama dengan populasi sebelumnya,

2. Probabilitas mutasi (Pm)

   Probabilitas mutasi menyatakan seberapa sering bagian-bagian kromosom akan dimutasikan. Jika tidak ada mutasi, keturunan diambil atau disalin langsung setelah *crossover* tanpa perubahan. Jika mutasi dilakukan, maka bagian-bagian tetentu kromosom diubah. jika probabilitas mutasinya 100%, keseluruhan kromosom diubah. Jika probabilitas mutasi 0%, maka tidak ada yang diubah. probabilitas mutasi dalam ruang algoritma genetika seharusnya diberi nilai yang kecil.

Parameter adalah parameter kontrol algoritma genetika, yaitu ukuran populasi (popsize), peluang *Crossover* (pc) dan peluang mutasi (pm). Rekomendasi untuk menentukan nilai parameter (Sri kusumadewi, 2003) :

1. Untuk permasalahan yang memiliki kawasan solusi cukup besar, De Jong merekomendasikan nilai parameter : (popsize; pc; pm) = (50;0.6;0.001)

2. Bila rata-rata *fitness* setiap generasi digunakan sebagai indikator, maka Grefenstette merekomendasikan : (popsize; pc; pm) = (30;0.95;0.01)
3. Bila *fitness* dari individu terbaik dipantau pada setiap generasi, maka usulannya adalah : (popsize; pc; pm) = (80;0.45;0.01)
4. Ukuran populasi sebaiknya tidak lebih kecil dari 30, untuk sembarang jenis permasalahan.

2.3 Penelitian Sebelumnya

Penelitian sebelumnya dilakukan Aulia Fitrah, Achmad Zaky, Fitrasani dengan judul Penerapan Algoritma Genetika Pada Persoalan Pedagang Keliling (TSP). Persoalan pedagang keliling (*Travelling Salesperson Problem-TSP*) merupakan persoalan optimasi untuk mencari perjalanan terpendek bagi pedagang keliling yang ingin berkunjung ke beberapa kota, dan kembali ke kota asal keberangkatan. Dengan metode algoritma genetika dapat membantu menemukan solusi untuk menentukan perjalanan terpendek yang melalui kota lainnya hanya sekali dan kembali ke kota asal keberangkatan.

Contoh Penyelesaian

Persoalan TSP yang diselesaikan dengan menggunakan algoritma genetika. Terdapat 5 kota yang akan dilalui oleh seorang pedangang keliling, misalnya Kota A,B,C,D,E. Perjalanan dimulai dari kota A dan berakhir di kota A.

Persoalan TSP akan diselesaikan dengan menggunakan algoritma genetika. Kriteria berhenti ditentukan yaitu apabila dalam beberapa generasi berturut-turut diperoleh nilai *fitness* yang terendah tidak berubah. Pemilihan nilai *fitness* yang terendah sebagai syarat karena nilai tersebut yang merepresentasikan jarak terdekat yang dicari pada persoalan TSP ini. Ada 4 kota yang akan menjadi gen dalam kromosom yaitu kota-kota selain kota asal.

a. Inisialisasi

Misalkan kita menggunakan 6 buah populasi dalam satu generasi, yaitu

Kromosom[1]= [B D E C]

Kromosom[2]= [D B E C]

Kromosom[3]= [C B D E]

Kromosom[4]= [E B C D]

Kromosom[5]= [E C B D]

Kromosom[6]= [C D E B]

b. Evaluasi kromosom

Kita akan menghitung nilai *fitness* dari tiap kromosom yang telah dibangkitkan:

Fitness[1] = AB+BD+DE+EC+CA

= 7 + 2 + 6 + 3 + 5 = 23

Fitness[2] = AD+DB+BE+EC+CA

= 9 + 2 + 8 + 3 + 5 = 27

Fitness[3] = AC+CB+BD+DE+EA

= 5 + 7 + 2 + 6 + 9 = 29

Fitness[4] = AE+EB+BC+CD+DA

= 9 + 8 + 7 + 4 + 9 = 37

Fitness[5] = AE+EC+CB+BD+DA

= 9 + 3 + 7 + 2 + 9 = 30

Fitness[6] = AC+CD+DE+EB+BA

= 5 + 4 + 6 + 8 + 7 = 30

c. Seleksi kromosom

Oleh karena pada persoalan TSP yang diinginkan yaitu kromosom dengan *fitness* yang lebih kecil akan mempunyai probabilitas untuk terpilih kembali lebih besar maka digunakan inverse.

Q[i]= $1/Fitness\ [i]$

Q[1]= $^1/_{23}$ =0.043

Q[2]= $^1/_{27}$ =0.037

Q[3]= $^1/_{29}$ =0.034

Q[4]= $^1/_{37}$ =0.027

Q[5]= $^1/_{30}$ =0.033

Q[6]= $^1/_{30}$ =0.033

Total= 0,043+0,037+0,034+0,027+0,033+0,033 = 0,207

Untuk mencari probabilitas kita menggunakan rumus berikut :

P[i]= $Q[1]/_{total}$

P[1]= $0.043/_{0.207}$ = 0.043

P[2]= $0.037/_{0.207}$ =0.037

P[3]= $0.034/_{0.207}$ =0.034

P[4]= $0.027/_{0.207}$ =0.027

P[5]= $0.033/_{0.207}$ =0.033

P[6]= $0.033/_{0.207}$ =0.033

Dari probabilitas di atas dapat terlihat bahwa kromosom ke-1 mempunyai *fitness* paling kecil mempunyai probabilitas untuk terpilih pada generasi selanjutnya lebih besar dari kromosom lainnya. Untuk proses seleksi kita menggunakan roulete- wheel, untuk itu kita terlebih dahulu mencari nilai kumulatif dari probabilitasnya.

C[1]= 0,028

C[2]= 0,028+0,179 = 0,387

C[3]= 0,387+0,164 = 0,551

C[4]= 0,551+0,130 = 0,681

C[5]= 0,681+0,159 = 0,840

C[6]= 0,840+0,159 = 1

Proses roulete-wheeladalah membangkitkan nilai acak R antara 0-1. Jika R[k]<C[k] maka kromosom ke-k sebagai induk, selain itu pilih kromosom ke-k sebagai induk dengan syarat C[k-1] < R[k] < C[k]. Kita putar roulete-wheel sebanyak jumlah kromosom yaitu 6 kali (membangkitkan bilangan acak R).

R[1]= 0,314

R[2]= 0,111

R[3]= 0,342

R[4]= 0,743

R[5]= 0,521

R[6]= 0,411

Setelah itu, populasi baru akan terbentuk yaitu :

Kromosom[1]= [2] = [D B E C]

Kromosom[2]= [1] = [B D E C]

Kromosom[3]= [3] = [C B D E]

Kromosom[4]= [5] = [E C B D]

Kromosom[5]= [4] = [E B C D]

Kromosom[6]= [6] = [C D E B]

d. *Crossover* (pindah silang)

Pindah silang pada TSP dapat diimplementasikan dengan skema order *Crossover*. Pada skema ini, satu bagian kromosom dipertukarkan dengan tetap menjaga urutan kota yang bukan bagian dari kromosom tersebut. Kromosom yang dijadikan induk dipilih secara acak dan jumlah kromosom yang di*Crossover* dipengaruhi oleh parameter *Crossover* probability (ρc). Misal kita tentukan ρc = 25%, maka diharapkan dalam 1 generasi ada 50% (3 kromosom) dari populasi mengalami *Crossover*. Pertama kita bangkitkan bilangan acak R sebanyak jumlah populasi yaitu 6 kali.

R[1]= 0,451

R[2]= 0,211

R[3]= 0,302

R[4]= 0,877

R[5]= 0,771

R[6]= 0,131

Kromosom ke-k yang dipilih sebagai induk jika R[k] < ρc. Maka yang akan dijadikan induk adalah kromosom[2], kromosom[3], dan kromosom[6]. Setelah melakukan pemilihan induk, proses selanjutnya adalah menentukan posisi *Crossover*. Hal tersebut dilakukan dengan membangkitkan bilangan acak antara 1 sampai dengan panjang kromosom-1. Dalam kasus TSP ini bilangan acaknya adalah antara 1-3. Misal diperoleh bilangan acaknya 1, maka gen yang ke-1 pada kromosom induk pertama diambil kemudian ditukar dengan gen pada kromosom induk kedua yang belum ada pada induk pertama dengan tetap memperhatikan urutannya. Bilangan acak untuk 3 kromosom induk yang akan di-*Crossover*:

C[2]= 2

C[3]= 1

C[6]= 2

Proses *Crossover*:

Kromosom[2]= Kromosom[2]>< Kromosom[3]

= [B D E C] >< [C B D E]

= [B D C E]

Kromosom[3]= Kromosom[3]>< Kromosom[6]

= [C B D E] >< [C D E B]

= [C D E B]

Kromosom[6]= Kromosom[6]>< Kromosom[2]

= [C D E B] >< [B D E C]

= [C D B E]

Populasi setelah di-*Crossover*:

Kromosom[1]= [D B E C]

Kromosom[2]= [B D C E]

Kromosom[3]= [C D E B]

Kromosom[4]= [E C B D]

Kromosom[5]= [E B C D]

Kromosom[6]= [C D B E]

e. Mutasi

Pada kasus TSP ini skema mutasi yang digunakan adalah *swapping mutation*. Jumlah kromosom yang mengalami mutasi dalam satu populasi ditentukan oleh parameter *mutation rate* (ρm). Proses mutasi dilakukan dengan cara menukar gen yang dipilih secara acak dengan gen sesudahnya. Jika gen tersebut berada di akhir kromosom, maka ditukar dengan gen yang pertama. Pertama kita hitung dulu panjang total gen yang ada pada satu populasi:

Panjang total gen = jumlah gen dalam 1 kromosom * jumlah Kromosom (3)

= 4 * 6 = 24

Untuk memilih posisi gen yang mengalami mutasi dilakukan dengan membangkitkan bilangan acak antara 1 – Panjang total gen yaitu 1- 24. Misal kita tentukan ρm = 20 %. Maka jumlah gen yang akan dimutasi adalah = 0,2*24 = 4,8 = 5

5 buah posisi gen yang akan dimutasi, setelah diacak adalah posisi 3, 7, 10, 20,24.

Proses mutasi :

Kromosom[1]= [D B C E]

Kromosom[2]= [B D E C]

Kromosom[3]= [C E D B]

Kromosom[4]= [E C B D]

Kromosom[5]= [D B CE]

Kromosom[6]= [E D B C]

Proses algoritma genetik untuk 1 generasi telah selesai. Maka nilai *fitness* setelah 1 generasi adalah:

Fitness[1] = AD+DB+BC+CE+EA

= 9 + 2 + 7 + 3 + 9 = 30

Fitness[2] = AB+BD+DE+EC+CA

= 7 + 2 + 6 + 3 + 5 = 23

Fitness[3] = AC+CE+ED+DB+BA

= 5 + 3 + 6 + 2 + 7 = 23

Fitness[4] = AE+EC+CB+BD+DA

= 9 + 3 + 7 + 2 + 9 = 30

Fitness[5] = AD+DB+BC+CE+EA

= 9 + 2 + 7 + 3 + 9 = 30

Fitness[6] = AE+ED+DB+BC+CA

= 9 + 6 + 2 + 7 + 5 = 29

Sebelumnya telah ditentukan kriteria berhenti yaitu bila setelah dalam beberapa generasi berturut-turut diperoleh nilai *fitness* yang terendah tidah berubah. Pada 1 generasi telah terlihat bahwa terdapat nilai *fitness* terkecil yang tidak berubah. Apabila perhitungan dilanjutkan hingga ke generasi ke-N maka diyakinkan bahwa nilai *fitness* yang terendah tetap tidak akan berubah. Walaupun perhitungan cukup dijabarkan hingga generasi ke-1 saja namun solusi yang mendekati optimal telah didapatkan. Oleh karena itu, terbukti bahwa algoritma genetika dapat menyelesaikan persoalan TSP