BAB 2

LANDASAN TEORI

2.1. Teka-Teki Silang

Teka-teki silang merupakan permainan sederhana yang banyak dimainkan dari berbagai kalangan. Cara bermain permaian ini memang sederhana, hanya merangkaikan jawaban soal dengan benar dan mengisikan jawabanya pada kotak kosong yang tersedia di papan teka-teki silang namun jawaban satu dengan yang lainnya harus saling berkaitan. Apabila satu jawaban salah maka akan sulit menemukan jawaban kata dari soal selanjutnya. Aturan permainan ini, kata yang dimasukkan minimal berjumlah tiga huruf, terdapat kata yang tersusun secara mendatar dan menurun dan kata yang tidak berkaitan itu dibatasi dengan kotak hitam. Pembuatan permainan ini dimulai dari mendesain papan teka-teki silang yang kemudian pembuat akan mencari sendiri jawaban yang cocok dengan keadaan papan teka-teki silang sehingga kata per kata dapat terangkai.

Permainan teka-teki silang ini pertama kali dikenalkan oleh Arthur Wynne pada tanggal 21 Desember 1913 [10]. Awalnya Arthur yang bekerja di sebuah media New York World diberikan tugas untuk membuat permainan yang menarik para pembaca. Suatu kali, ia teringat pada masa kecilnya. Arthur ingat bahwa ia pernah memainkan sebuah permainan yang dinamakan "Magic Squares". Permainan itu adalah permainan kata-kata, dimana sang pemain harus menyusun kata agar sama mendatar dan menurun hingga membentuk kotak. Dari permainan ini, ia kemudian mencoba berkreasi dengan menambah luasan kata-kata dengan bentuk yang lebih kompleks dan untuk menyusun hal itu, ia memberi semacam pertanyaan untuk membuka kunci jawabannya. Kemudian, pada tahun 1942-an, New York Times, koran ternama di Amerika membuat semacam standar untuk TTS [5]. Standar itu seperti bentuk yang simetris dan panjang kata minimal tiga huruf. Hal ini membuat permainan TTS makin digemari dan populer, hingga akhirnya menyebar ke berbagai belahan dunia.

2.2. Algoritma

Di bidang komputerisasi atau matematika, algoritma merupakan perintah untuk menyelesaikan suatu permasalahan atau pengambilan keputusan di mana permasalahan yang akan diambil mempunyai kondisi awal yang harus dipenuhi sebelum menjalankan algoritma. Seorang analisis sistem tentunya menggunakan menggunakan algoritma untuk merancang suatu sistem dan bagi seorang programmer, algoritma digunakan untuk membuat modul-modul program.

2.2.1. Definisi Algoritma

Algoritma merupakan suatu urutan langkah-langkah dalam memecahkan suatu permasalahan. Algoritma harus dibuat secara sistematis agar komputer dapat mengerti dan mengeksekusinya dengan benar. Algoritma mempunyai tiga komponen yaitu masukan, proses dan keluaran. Komponen masukan terdiri dari pemilihan variable, jenis variable, tipe variable, konstanta dan parameter ke dalam suatu fungsi. Komponen proses merupakan bagian utama dan terpenting dalam merancang sebuah algoritma. Dalam komponen ini terdapat logika algoritma, rumusan, dan metode. Komponen keluaran merupakan tujuan perancangan algoritma dan program . Permasalahan yang diselesaikan dalam komponen proses harus ditampilakn dalam komponen keluaran.

2.2.2. Sejarah Algoritma

Dilihat dari asal-usul katanya Algoritma mempunyai sejarah tersendiri. Orang hanya menemukan kata *Algorism* yang berarti proses menghitung dengan angka arab. Para ahli bahasa berusaha menemukan asal kata ini namun hasilnya kurang memuaskan. Akhirnya para ahli sejarah matematika menemukan asal kata tersebut yang berasal dari nama penulis buku arab yang terkenal yaitu Abu Ja’far Muhammad Ibnu Musa Al-Khuwarizmi. Al- Khuwarizmi dibaca orang barat menjadi *Algorism.* Al-Khuwarizmi menulis buku yang berjudul *Kitab Al Jabar Wal-Muqabala* yang artinya “Buku pemugaran dan pengurangan” (*The book of restoration and reduction*) [9]. Dari judul buku itu kita juga memperoleh akar

kata “Aljabar” (*Algebra*). Perubahan kata dari *Algorism* menjadi *Algorithm* muncul karena kata *Algorism* sering dikelirukan dengan *Arithmetic*, sehingga akhiran –*sm* berubah menjadi –*thm.* Karena perhitungan dengan angka Arab sudah menjadi hal yang biasa. Maka lambat laun kata *Algorithm* berangsur-angsur dipakai sebagai metode perhitungan (komputasi) secara umum, sehingga kehilangan makna kata aslinya. Dalam Bahasa Indonesia, kata *Algorithm* diserap menjadi *Algoritma*.

2.3. Algoritma Genetika

Algoritma ini pertama kali diperkenalkan oleh John Holland pada tahun 1975 dari Universitas Michigan dalam bukunya yang berjudul “*Adaption in Natural and Artificial System*” [4], kemudian pada tahun 1989 dikembangkan oleh muridnya yang bernama David Goldberg. John Holland menyatakan apabila masalah yang berbentuk adaptasi alami maupun buatan dapat diformulasikan ke dalam terminology genetika, lalu muridnya menyatakan bahwa algoritma genetika ini sebagai suatu pencarian algortima berdasarkan mekanisme seleksi dan genetika alam. Algoritma genetika terispirasi dari mekanisme seleksi alam, di mana individu yang lebih kuat yang akan menjadi pemenang dalam lingkungan kompetitif dan solusi yang optimal dapat diperoleh dan diwakilkan oleh pemenang akhir dari permainan genetika.

2.3.1. Struktur Algoritma Genetika

Algoritma genetika melakukan teknik pencarian secara sekaligus atas sejumlah solusi yang dikenal dengan istilah populasi sedangkan individu yang terdapat pada sebuah populasi disebut dengan kromosom. Penentuan populasi awal didapat secara acak yang kemudian populasi berikutnya akan didapat dari hasil proses genetika.

Ada beberapa hal yang harus dilakukan dalam algoritma genetika adalah :

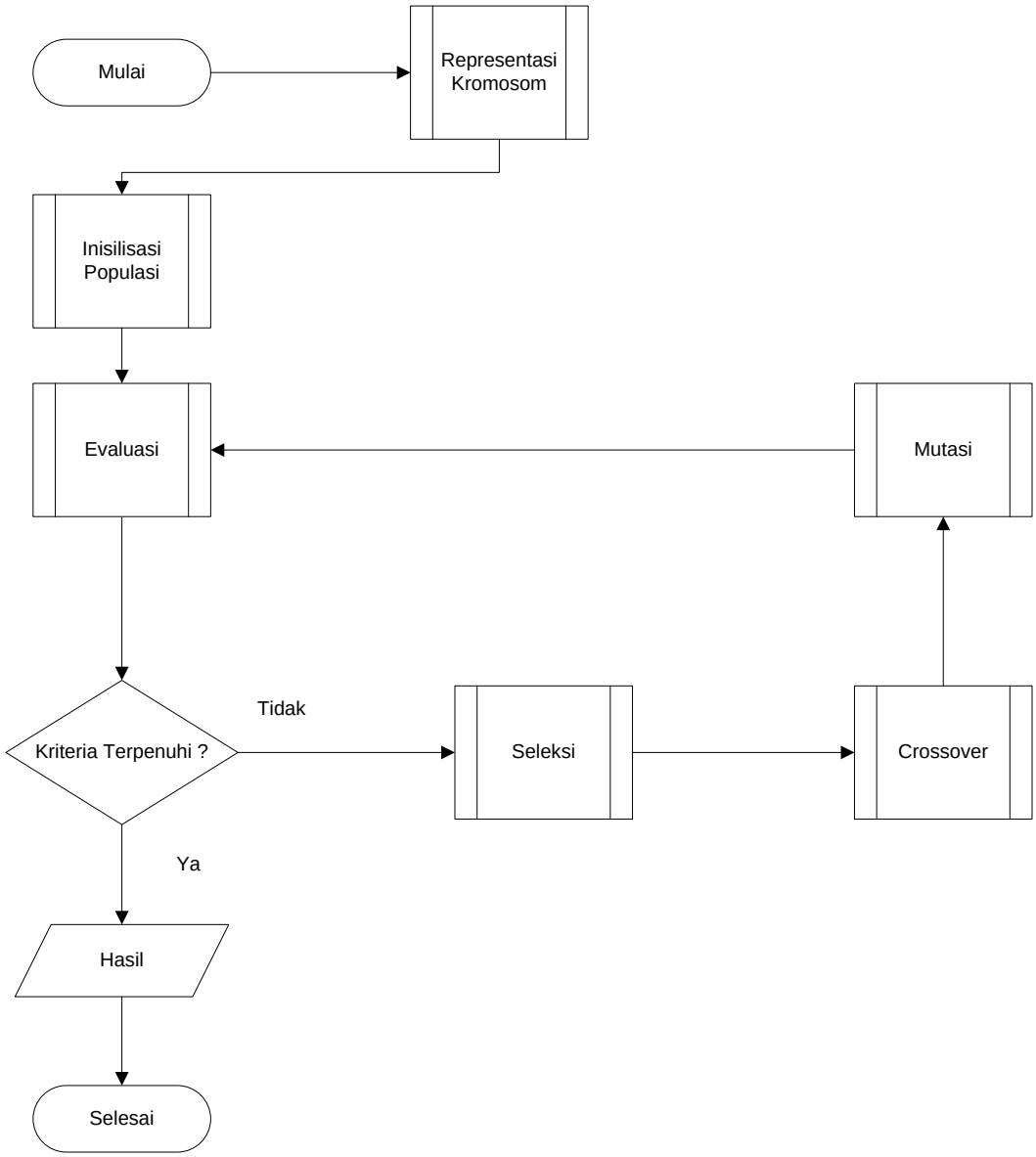
1. Mendefinisikan individu, merupakan pernyataan solusi awal dari kasus yang diambil

2. Mendefinisikan nilai fitness, merupakan suatu ukuran seberapa baik individu dalam mencapai solusi
3. Menentukan proses pembangkitan populasi awal yang dilakukan secara acak
4. Menentukan proses seleksi, *crossover* atau perkawinan silang.

Beberapa definisi penting yang digunakan dalam Algoritma Genetika, antara lain :

a. *Genotype* (Gen), sebuah nilai yang menyatakan satuan dasar yang membentuk arti tertentu dalam satu kesatuan yang disebut kromoson. Gen ini dapat berupa nilai biner, float, integer maupun karakter.
b. Kromoson, gabungan gen yang membentuk nilai tertentu.
c. Individu, menyatakan suatu nilai atau keadaan yang menyatakan salah satu solusi yang mungkin dari permasalahan yang diangkat.
d. Populasi, sekumpulan individu yang akan diproses bersama dalam satu siklus proses evolusi.
e. Nilai *Fitness*, menyatakan seberapa baik nilai dari suatu individu atau solusi yang didapatkan.

Algoritma genetika mempunyai beberapa komponen di dalam prosesnya yaitu representasi kromosom, prosedur inisialisasi, fungsi evaluasi, seleksi, operator genetika dan penentuan parameter. Berikut ini pada Gambar 2-1 adalah *flowchart* alur proses dari algoritma genetika :



Gambar 0-1 *Flowchart* algoritma genetika

1. Representasi Kromosom

Suatu permasalahan yang akan dikonversikan dulu ke dalam suatu individu yang diwakili oleh satu atau lebih kromosom dengan kode tertentu hal ini agar dapat diproses dengan menggunakan algoritma genetika [5].

Representasi kromosom meliputi proses pengkodean gen dari kromosom. Contoh dari representasi kromosom antara lain sebagai berikut :

1. String bit : 11111, 1001

2. Bilangan Real : 24.01, 404.24
3. Elemen Permutasi: E2, E10
4. Daftar Aturan : R1, R2, R3
5. Elemen Program : pemrograman genetika
6. Struktur lainnya

Gen adalah bagian dari kromosom, dimana satu gen akan mewakili satu variabel. Selain itu gen dapat direpresentasikan dalam bentuk string bit, pohon, array bilangan real, daftar aturan, elemen permutasi, elemen program dan lain-lain. Dalam hal ini akan ditentukan yang mana gen dan yang mana kromosom ke permasalahan yang ada. Seperti pada permasalahan teka-teki silang, yang diketahui sebagai gen adalah kata atau calon jawaban yang dimasukan oleh *user* sedangkan yang disebut dengan kromosom adalah kumpulan dari kata yang dimasukkan oleh *user* karena telah disebutkan sebelumnya bahwa kromosom adalah kumpulan dari gen.

2. Prosedur Inisialisasi

Proses inisialisasi terhadap kromosom ini sebelumnya menentukan ukuran populasi di mana ukuran populasi ini tergantung pada permasalahan yang akan dipecahkan dan jenis operator genetika yang diimplementasikan. Apabila ukuran populasi terlalu besar maka komputasi yang dibutuhkan juga akan lebih besar, sehingga waktu yang dibutuhkan juga akan bertambah. Namun apabila ukuran populasi terlalu kecil maka alternatif solusi akan sedikit, sehingga kemungkinan hasil yang didapatkan kurang baik. Dengan pertimbangan tersebut maka perlu menentukan ukuran populasi yang tidak terlalu besar atau terlalu kecil. Ukuran populasi sebaiknya tidak lebih kecil dari 30 untuk setiap jenis permasalahan [2]. Inisialisasi kromosom dilakukan secara acak dengan memperhatikan permasalahan yang ada. Dari proses inisialisasi ini akan menghasilkan solusi awal yang akan diambil untuk proses berikutnya. Dalam menginisialisasi populasi sehingga menghasilkan solusi awal ini tergantung dengan permasalahan yang akan dipecahkan dengan memperhatikan solusi yang akan dicapai. Sebagai

contoh dalam permasalahan teka-teki silang solusi yang ingin dicapai adalah menghasilkan kata-kata yang terangkai.

3. Fungsi Evaluasi / Fungsi Fitnes

Untuk menentukan fungsi fitness yang tepat untuk suatu permasalahan pertama yang perlu dilakukan adalah memperhatikan fungsi objektif. Dalam fungsi ini mengenal dua masalah yaitu maksimasi atau minimasi. Maksimasi maksudnya adalah mencari nilai maksimal dari sesuatu (bias berupa fungsi) sehingga tujuannya adalah memaksimalkan suatu kondisi bias fungsi ataupun yang lainnya sedangkan minimasi itu kebalikan dari maksimasi.

Dalam permasalahan yang diambil itu adalah untuk mencari nilai fitness terbesar sehingga dalam hal ini disebut maksimasi. Penentuan fungsi fitness ini berpengaruh terhadap solusi yang akan dihasilkan. Dengan mempertimbangkan dengan permasalahan yang ada fungsi fitness yang akan digunakan untuk proses analisis yaitu sebagai berikut :

$$f = \frac{\left(\sum_{j=0}^{n} \boldsymbol{gen[i]}\right)}{tkp\ X\ loop} \quad (1)$$

Keterangan :

f = nilai fitness

gen[i] = nilai gen ke-i

tkp = total kata terpakai

loop = percobaan ke-n (n adalah jumlah proses pengulangan)

sedangan untuk mencari hasil dari gen[j] adalah sebagai berikut :

$$\boldsymbol{gen[i]} = \frac{\boldsymbol{tp}}{\boldsymbol{tk}} \quad (2)$$

Keterangan :

tp = jumlah titik perpotongan yang ada di gen ke-i

tk = jumlah total karakter yang ada di gen ke-i

4. Seleksi

Proses seleksi ini akan menentukan individu-individu mana saja yang akan dipilih untuk dilakukan rekombinasi. Ada beberapa metode seleksi yang dapat digunakan, antara lain :

a) Rank-based fitness assignment

Populasi diurutkan menurut nilai objektifnya [12]. Nilai fitness dari tiap-tiap individu hanya tergantung pada posisi individu tersebut dalam urutan, dan tidak dipengaruhi oleh nilai objektifnya. Sebagai contoh di bawah ini terdapat tabel 2-1 dengan kromosom beserta nilai *fitness*:

Tabel 0-1 Contoh Kromosom Rank Base

Kromosom	Fitness
A	15
B	5
C	10
D	5
E	6

Setelah proses pengurutan dan pemberian nilai fitness baru, setiap kromosom akan memiliki kesempatan yang lebih adil untuk terpilih. Tabel 2-2 di bawah ini menunjukan bahwa kromosom pada tabel 2-1 telah diurutkan sesuai dengan nilai *fitness* :

Tabel 0-2 Contoh Hasil Rank Base

Kromosom	Fitness	Fitnes Baru
B	5	1
D	5	2
E	6	3
C	10	4
A	15	5

b) Roulette wheel selection

Pada metode ini, orang tua dipilih berdasarkan nilai fitnessnya, semakin baik nilai fitnessnya maka semakin besar kemungkinannya untuk terpilih. Diandaikan semua kromosom diletakkan pada sebuah roda roulette, besarnya kemungkinan bagi setiap kromosom adalah tergantung dari nilai fitnessnya.

Seleksi ini bertujuan untuk memberikan kesempatan reproduksi yang lebih besar bagi anggota populasi yang memiliki fitness tinggi untuk melakukan reproduksi. Dibawah ini merupakan algoritma dari *roulette wheel selection* :

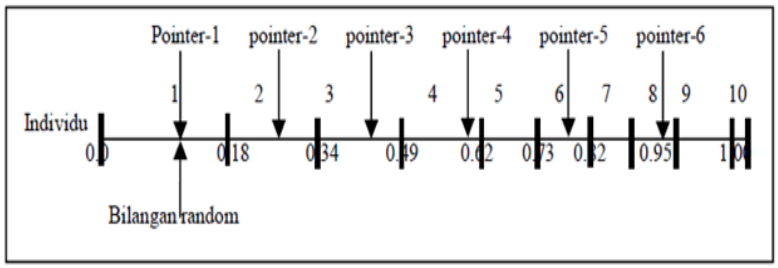
1) Dihitung nilai fitness masing-masing individu (f_i, dimana i adalah individu ke 1 s/d ke-n)
2) Dihitung total fitness semua individu
3) Dihitung *fitness* relatif masing-masing individu
4) Dari *fitness* relatif tersebut, dihitung fitness kumulatifnya.
5) Dibangkitkan nilai random
6) Dari bilangan random yang dihasilkan, ditentukan individu mana yang terpilih dalam proses seleksi

Skema seleksi dengan roda roulette ini adalah berdasarkan skala fitness (*fitness scale*). Karena terpilihnya suatu kromosom dalam populasi untuk dapat berkembang biak adalah sebanding dengan *fitness* tersebut.

c) Stochastic universal sampling

Pada metode ini, individu-individu dipetakan dalam suatu segmen garis secara berurutan sedemikian hingga tiap-tiap segmen individu memiliki ukuran yang sama dengan ukuran fitness-nya seperti halnya pada seleksi roda roulette, dan diberikan sejumlah pointer sebanyak individu yang diseleksi di garis tersebut [12]. *Stochastic universal sampling* memiliki nilai bias nol dan penyebaran yang minimum. Pada metode ini, individu-individu dipetakan dalam suatu segmen garis secara berurutan sedemikian hingga tiap-tiap segmen individu memiliki ukuran yang sama dengan ukuran *fitness*nya seperti halnya pada seleksi roda roulette, dan diberikan sejumlah *pointer* sebanyak individu yang diseleksi di garis tersebut. Andaikan N adalah jumlah individu, dan posisi pointer pertama diberikan secara acak pada range [1, 1/N]. Apabila ada 6

individu yang akan diseleksi, maka jarak antar *pointer* adalah 1/6=0.167 (gambar 1.2), sehingga misalkan bilangan random yang dibangkitkan pada range [0, 0.167] adalah 0, 1, maka hasil yang diperoleh setelah seleksi adalah :



Gambar 0-2 *Stohastic Universal Sampling*

d) Tournament selection

Pada metode seleksi dengan turnamen ini, akan 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 [12]. Parameter yang digunakan pada metode ini adalah ukuran tour yang bernilai antara 2 sampai N (jumlah individu dalam suatu populasi). Pada seleksi alam yang terjadi di dunia nyata, beberapa individu (biasanya individu jantan) berkompetisi dalam sebuah kelompok kecil sampai tersisa hanya satu individu pemenang. Individu pemenang inilah yang bisa kawin (pindah silang). Metode *roulette-wheel selection* tidak mengkombinasikan masalah ini. Sebuah metode *tournament selection* mencoba mengadopsi karakteristik alami ini. Dalam bentuk paling sederhana, metode ini mengambil dua kromosom secara random dan kemudian menyeleksi salah satu yang bernilai *fitness* paling tinggi untuk menjadi orang tua pertama. Cara yang sama dilakukan lagi untuk mendapatkan orang tua yang kedua.

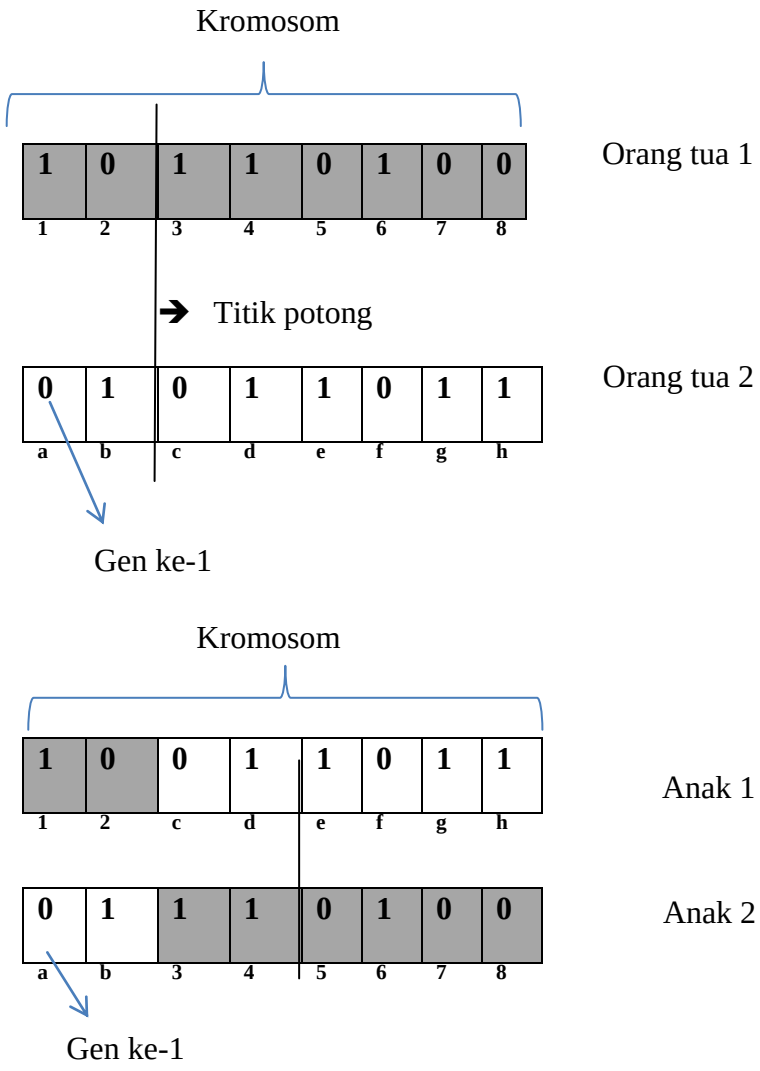
5. Operator Genetika

a) *Crossover*

Operator algoritma yang melakukan proses pindah silang sehingga menghasilkan individu baru. Kromosom dari salah satu individu akan bertukar dengan kromosom dari individu yang lain.

1. *Crossover* satu titik

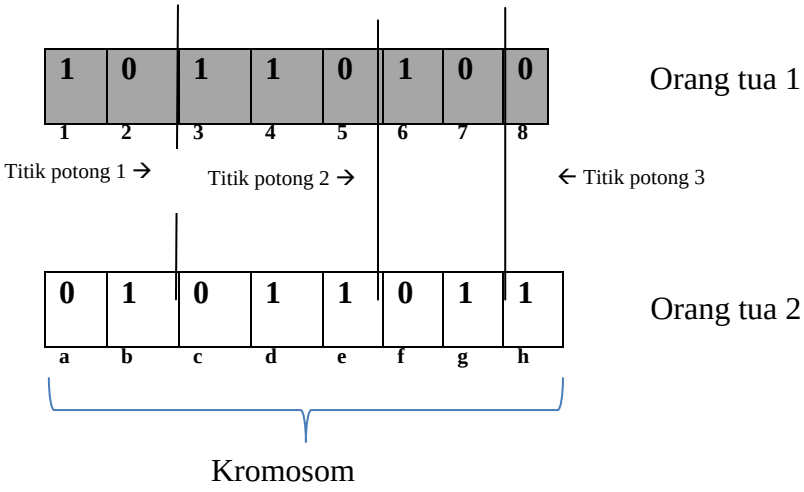
Sebuah titik potong *crossover* dipilih kemudian dari hasil titik potong dari salah satu kromosom saling tukar ke titik potong kromosom yang lainnya sehingga menghasilkan kromosom yang baru. Titik potong bias ditentukan di mana saja hanya saja yang harus dipertimbangkan adalah jumlah titik kromosom orang tua 1 dengan orang tua 2 harus sama. Pada gambar 2-3 dapat dilihat proses *crossover* satu titik di bawah ini :

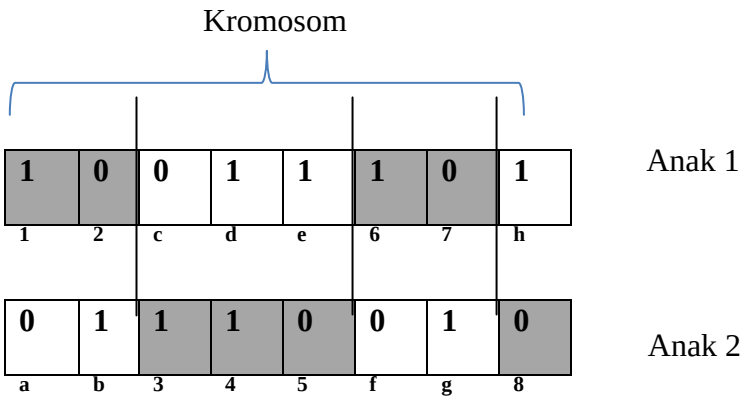


Gambar 0-3 Proses *Crossover* Satu Titik

2. *Crossover* banyak titik

Seperti *crossover* satu titik hanya saja titik potongnya lebih dari satu. Titik potong bisa ditentukan di mana saja hanya saja yang harus dipertimbangkan adalah jumlah titik kromosom orang tua 1 dengan orang tua 2 yang dipotong harus sama. Proses *crossover* banyak titik dapat dilihat pada gambar 2-4 di bawah ini :





Gambar 0-4 Proses Crossover Satu Titik

b) Mutasi

Operator genetika selain *crossover* ini menghasilkan perubahan secara acak pada kromosom. Operator ini akan mengubah bit yang awalnya 0 menjadi 1 sedangkan bit awal yang awalnya bernilai 1 akan diganti menjadi 0 sehingga pada mutasi ini memungkinkan muncul kromosom baru yang sebelumnya tidak ada di populasi awal.

6. Penentuan Parameter

Nilai dari parameter ditentukan berdasarkan permasalahan yang akan dipecahkan yang digunakan sebagai parameter kontrol algoritma genetika. Parameter kontrol algoritma genetika adalah ukuran populasi (popsize), peluang crossover (p_c) dan peluang mutasi (p_m).

2.4. Algoritma Backtracking

Algoritma *backtracking* merupakan algoritma yang berbasis pada *Depth First Search* (DFS), tetapi hanya pencarian yang mengarah ke solusi saja yang dipertimbangkan. Artinya, jika dalam pencarian menemui langkah yang tidak mengarah ke solusi, maka akan dicari langkah yang lain. Langkah-langkahnya adalah sebagai berikut [11] :

a. Solusi dicari dengan membentuk lintasan dari akar ke daun. Aturan pembentukan yang dipakai adalah aturan dalam metode DFS. Simpul-simpul yang sudah dilahirkan dinamakan simpul hidup (*live node*). Simpul hidup yang sedang diperluas dinamakan simpul-E (*expand node*). Simpul dinomori dari atas ke bawah sesuai dengan urutan kelahirannya.
b. Tiap kali simpul-E diperluas, lintasan yang dibangun olehnya bertambah panjang. Jika lintasan yang sedang dibentuk tidak mengarah ke solusi, maka simpul-E tersebut “dibunuh” sehingga menjadi simpul mati (*dead node*). Fungsi yang digunakan untuk membunuh simpul-E adalah dengan menerapkan fungsi pembatas (*bounding function*). Simpul yang sudah mati tidak akan pernah diperluas lagi.

c. Jika pembentukan lintasan berakhir pada simpul mati, maka proses pencarian diteruskan dengan membangkitkan simpul anak yang lainnya. Bila tidak ada lagi simpul anak yang dapat dibangkitkan, maka pencarian solusi dilanjutkan dengan melakukan *backtrack* ke simpul hidup terdekat (simpul orang tua). Selanjutnya simpul ini menjadi simpul-E yang baru. Lintasan baru dibangun kembali sampai lintasan tersebut membentuk solusi.
d. Pencarian dihentikan bila solusi ditemukan atau tidak ada lagi simpul hidup untuk *backtrack*.

2.5. Big-O

Notasi *Big O* merupakan suatu notasi matematika untuk menjelaskan batas atas dari *magnitude* suatu fungsi dalam fungsi yang lebih sederhana. Dalam dunia ilmu komputer, notasi ini sering digunakan dalam analisis kompleksitas algoritma. Notasi *Big O* pertama kali diperkenalkan pakar teori bilangan Jerman, Paul Bachman tahun 1894, pada bukunya yang berjudul *Analytische Zahlentheorie* edisi kedua. Notasi tersebut kemudian dipopulerkan oleh pakar teori bilangan Jerman lainnya, Edmund Landau, dan oleh karena itu, terkadang disebut sebagai symbol Landau.

Untuk membandingkan kompleksitas algoritma yang satu dengan yang lain, dapat digunakan tabel 2-1 jenis kompleksitas, yang diurutkan berdasarkan kompleksitas yang paling baik ke yang paling buruk. Sebuah masalah yang mempunyai algoritma dengan kompleksitas polinomial kasus-terburuk dianggap mempunyai algoritma yang “bagus” artinya masalah tersebut mempunyai algoritma yang mangkus, dengan catatan polinomial tersebut berderajat rendah. Jika polinomnya berderajat tinggi, waktu yang dibutuhkan untuk mengeksekusi algoritma tersebut panjang [1].

Tabel 0-3 Jenis Kompleksitas

Notasi	Nama
$O(1)$	Konstan

$O(\log^* n)$	Logaritma iterative
$O(\log n)$	Logaritmik
$O([\log n]^c)$	Polilogaritmik
$O(n)$	Linier
$O(n \log n)$	Linierithmik, loglinier, quasilinier or supralinier
$O(n^2)$	Kuadratik
$O(n^c)$, $c > 1$	Polinomial (kadang disebut *algebraic*)
$O(c^n)$	Eksponensial (kadang disebut *geometric*)
$O(n!)$	Faktorial, kombinatorial

Sebagai contoh terdapat suatu program sederhana yang akan dihitung komplesitasnya dengan menggunakan Big-O seperti :

```
Sum = 0;
For (i=0; i<n; i++)
Sum= sum +a[i]
```

Kemudian pada tabel 2-4 dapat dilihat perhitungan kompleksitas dengan menggunakan Big-O dari contoh program sederhana di atas :

Tabel 0-4 Contoh Perhitungan Big-O dengan program sederhana

Program	Notasi Big-O	Keterangan
`Sum = 0`	O(1)	Dieksekusi 1 kali
`i=0`	O(1)	Dieksekusi 1 kali
`i<n`	O(N)	Dieksekusi n kali
`i++`	O(N)	Dieksekusi n kali
`Sum= sum +a[i]`	O(N)	Dieksekusi n kali

Dari hasil tabel 2-4 di atas maka dapat dihitung hasil dari notasi Big-O adalah sebagai berikut :

$O(1) + O(1) + O(N) + O(N) + O(N) = O(N)$

Jadi dari program sederhana di atas mempunyai hasil perhitungan dengan Big-O adalah O(N) jadi untuk waktu eksekusinya sebanding dengan jumlah data jika n= 5 maka waktu eksekusinya pun 5.