BAB II
LANDASAN TEORI

2.1 Travelling Salesman Problem (TSP)

Persoalan TSP merupakan salah satu persoalan optimasi *kombinatorial* (kombinasi permasalahan). Banyak permasalahan yang dapat direpresentasikan dalam bentuk TSP. Persoalan ini sendiri menggunakan representasi graf untuk memodelkan persoalan yang diwakili sehingga lebih memudahkan penyelesaiannya. Diantara permasalahan yang dapat direpresentasikan dengan TSP adalah pencarian rute bis sekolah untuk mengantarkan siswa, pengambilan tagihan telepon, efisiensi pengiriman surat atau barang, perancangan pemasangan pipa saluran dan lain-lain. Persoalan yang muncul adalah bagaimana cara mengunjungi node (simpul) pada graf dari titik awal ke setiap titik-titik lainnya dengan bobot minimum (biaya paling murah) dan kembali lagi ke asal node. Bobot atau biaya ini sendiri dapat mewakili berbagai hal, seperti berapa biaya minimum, jarak minimum, bahan bakar minimal, waktu minimum, kenyamanan dan lain-lain.

Pada TSP, jumlah jalur yang mungkin diperoleh dengan menggunakan rumus permutasi berikut ini:

$$P_n^k = \frac{n!}{(n-k)!} \quad \text{.....(2.1)}$$

dimana *n* adalah jumlah seluruh node dan *k* adalah jumlah node yang diseleksi.

Terdapat dua jenis TSP, yaitu:

1. TSP asimetris, biaya dari node 1 ke node 2 tidak sama dengan biaya dari node 2 ke node 1.

$$n12 \neq n21 \text{.....(2.2)}$$

Jumlah jalur yang mungkin adalah permutasi dari jumlah node dibagi dengan jumlah node. Hal ini dapat dipahami secara siklus, sebuah jalur dengan urutan $n123 = n231 = n312$. Tetapi jalur dengan urutan $n123 \neq n321$. Jadi apabila terdapat 10 node, maka jalur yang mungkin untuk TSP asimetris adalah:[1]

$$P_{10}^{10} = \frac{10!}{10\times(10-10)!} = 362.880 \text{.....(2.3)}$$

2. TSP simetris

$$n12 = n21 \quad \text{.....} \quad (2.4)$$

Jumlah jalur yang mungkin adalah permutasi dari jumlah node dibagi dengan dua kali jumlah node. Hal ini dapat dipahami karena secara siklus sebuah jalur dengan urutan $n123 = n231 = n312$. Karena biaya dari $n12 = n21$, maka jalur dengan urutan $n123 = n321$. Jadi apabila terdapat 10 node, maka jalur yang mungkin untuk TSP simetris adalah:[1]

$$P_{10}^{10} = \frac{10!}{2x10x(10-10)!} = 181.440 \quad \text{.....} \quad (2.5)$$

2.1.1 Sejarah Travelling Salesman Problem

Permasalahan matematika tentang *Traveling Salesman Problem* dikemukakan pada tahun 1800 oleh Matematikawan Irlandia William Rowan Hamilton dan Matematikawan Inggris Thomas Penyngton. Diskusi mengenai awal studi dari Hamilton dan Kirkman dapat ditemukan di *Graph Theory* tahun *1736-1936* oleh N. L. Biggs, E. K. LLoyd, dan R. J. Wilson, di Oxford, tahun 1976.

Bentuk umum dari TSP pertama dipelajari oleh para Matematikawan mulai tahun 1930. Diawali oleh Karl Menger di Vienna dan Harvard. Setelah itu permasalahan TSP dipublikasikan oleh Hassler Whitney dan Merrill Flood di Princeton. Penelitian secara detail dari hubungan antara Menger dan Whitney, dan perkembangan TSP sebagai sebuah topik studi dapat ditemukan pada makalah Alexander Schrijver's "*On the history of combinatorial optimization (*sejak *1960)*".[2]

2.1.2 Contoh Perkembangan Masalah yang Muncul

Kode program komputer yang dibuat untuk menyelesaikan persoalan TSP telah berkembang semakin baik dari tahun ke tahun. Tanda yang paling mencolok dari perkembangan metode untuk menyelesaikan persoalan TSP adalah bertambahnya jumlah simpul (*node*) yang dapat diselesaikan, mulai dari solusi persoalan 49 kota yang dikembangkan oleh Dantzig, Fulkerson, dan Johnson's pada tahun 1954 sampai kepada solusi persoalan 24.978 kota pada tahun 2004. Data-data tersebut didapat dari National Imagery and Mapping Agency, sebuah database nasional yang menyimpan nama-nama fitur geografi.

2.2 Algoritma Genetika (AG)

Algoritma genetika adalah algoritma komputasi yang terinspirasi dari teori evolusi yang kemudian diadopsi menjadi algoritma komputasi untuk mencari solusi suatu permasalahan dengan cara yang lebih “alamiah”. Salah satu aplikasi algoritma genetika adalah pada permasalahan optimasi kombinasi, yaitu mendapatkan suatu nilai solusi optimal terhadap suatu permasalahan yang mempunyai banyak kemungkinan solusi. Ahli biologi yang ingin mensimulasikan proses evolusi, kemudian mengajak ahli komputer untuk merealisasikannya.

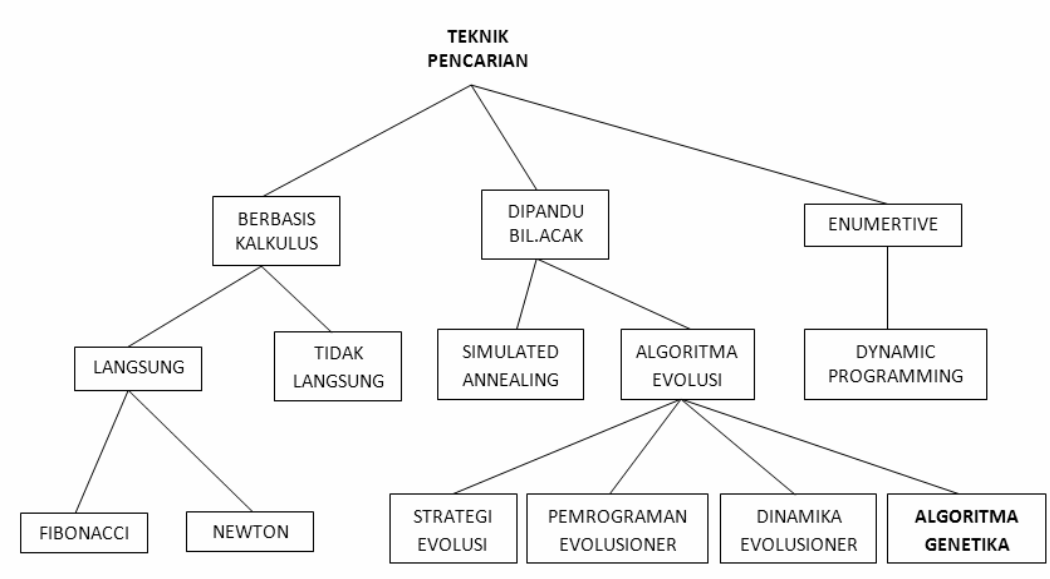
Algoritma Genetika diusulkan pertama kali oleh John Holland dan teman-temannya di universitas Michigan untuk aplikasi seluler otomata pada tahun 1960-an. John Holland mengatakan bahwa setiap masalah yang berbentuk adaptasi (alamiah maupun buatan) dapat diformulasikan dalam terminologi genetika. Algoritma Genetika adalah simulasi dari proses evolusi Darwin dan operasi genetika atas kromosom. Teknik ini menjadi populer di kalangan ilmuwan dan rekayasawan seluruh dunia untuk memecahkan masalah optimasi mereka. Aplikasinya antara lain meliputi *job shop scheduling*, pembelajaran pengendali *neuro-fuzzy*, pemprosesan citra dan optimasi kombinatorial.

Algoritma Genetika secara khusus dapat diterapkan untuk memecahkan masalah optimasi yang kompleks. Karena itu Algoritma Genetika baik untuk aplikasi yang memerlukan strategi pemecahan masalah secara adaptif. Algoritma Genetika mencari pemecahan (solusi) yang ‘terbaik’ dapat dilakukan melalui struktur genetik (John Holland menyebutnya sebagai *building block*) yang menyatakan sejumlah kemungkinan penyelesaian.[2]

Algoritma Genetika adalah teknik optimasi yang terkenal. Secara umum ada tiga golongan besar teknik optimasi yaitu:

1. Berbasis pada kalkulus.
2. Berbasis pada enumeriatif atau perulangan.
3. Berbasis pada pencarian acak terarah (*guide random search*).

Seperti dapat dilihat pada gambar berikut ini.



Gambar 2.1 Diagram pengelompokan dalam teknik pencarian

Setelah John Holland memaparkan idenya tentang Algoritma Genetika maka banyak peneliti yang mengusulkan beberapa variasi dari algoritma dasar Algoritma Genetika. Tetapi ciri penting dari seluruh Algoritma Genetika adalah teknik penanganan populasinya. Ide awal Algoritma Genetika mengadopsi kebijakan penggantian umum dimana keseluruhan populasi diganti pada setiap generasi. Di pihak lain, kebijakan keadaan tunak (*steady-state policy*) menerapkan penggantian selektif bagi populasi. Hal ini lebih alami karena di alam sudah biasa orang tua dan anak hidup berdampingan pada saat yang bersamaan. Kebijakan ini banyak diaplikasikan pada banyak variasi Algoritma Genetika.

Penyajian individu dari Algoritma Genetika biasanya menggunakan sekumpulan bilangan biner. Tetapi penyajian bilangan biner ini memiliki beberapa kelemahan bila diaplikasikan pada masalah multidimensional dan persoalan numerik persisi. Karena itu, D.E Golberg mengusulkan pemakaian bilangan nyata atau floating-point Z. Mischalewisz menunjukkan bahwa penyajian dengan cara demikian mampu mengalahkan Algoritma Genetika dengan penyajian biner, dalam hal kebugaran (*fitness*) dan waktu komputasi CPU (*Central Processing Unit*). Sejak itu banyak operator khusus seperti *arithmatical crossover* dan mutasi tidak seragam diusulkan.

Pada dasarnya ada 4 kondisi yang sangat mempengaruhi proses evaluasi, yaitu:

1. Kemampuan organisme untuk melakukan reproduksi.
2. Keberadaan populasi organisme yang bisa melakukan reproduksi.
3. Keberagaman organisme dalam suatu populasi.
4. Perbedaan kekuatan dan kemampuan organisme untuk bertahan hidup.

Individu yang lebih kuat (*fit*) akan memiliki tingkat *survival* atau tingkat daya bertahan hidup yang lebih tinggi. Selain itu individu yang semakin kuat akan memiliki tingkat reproduksi yang lebih tinggi jika dibandingkan dengan individu yang kurang *fit*. Pada kurun waktu tertentu (sering dikenal dengan istilah generasi), populasi secara keseluruhan akan memuat lebih banyak organisme yang *fit*.

2.2.1 Sejarah Algoritma Genetika

Teori Darwin yang sempat membuat orang orang berfikir bahwa manusia berasal dari kera, bahkan lebih rendah telah membutakan kita sekitar abad 19 hingga beberapa tahun belakangan ini. Pada abad ke 19 banyak ilmuwan yang mencoba untuk membuktikan dan mensimulasikannya. Neo darwinisme yang menyebutkan bahwa sejarah kehidupan mahkluk hidup adalah melalui suatu mekanisme proses statistika yang terjadi antara populasi dan spesies, yang dikenal dengan proses manipulasi genetika. Proses ini masing-masing adalah **reproduksi, mutasi, kompetisi dan pemilihan.**

Cikal bakal penggunaan Algoritma Genetika untuk pencarian dalam sistem buatan diprakarsai oleh beberapa ahli biologi yang menggunakan komputer digital untuk mengerjakan simulasi dari sistem genetika. Diantara para ahli tersebut adalah:[2]

1. Baricelli, N.A pada tahun 1957 melakukan penelitian tentang proses evolusi simbiogenetik yang direalisasikan dengan sistem artificial.
2. Fraser, A.S pada tahun 1960 mensimulasikan sistem genetika dengan komputer, yang meliputi aspek-aspek S-linkage, dominasi dan epistasis.
3. Baricelli, N.A pada tahun 1962 mengajukan teory evolusi dan analisis numeriknya.

Meskipun penelitian-penelitian tersebut bertujuan untuk meneliti gejala alam namun yang mereka kerjakan secara kebetulan memiliki pemikiran paralel yang memunculkan ide tentang Algoritma Genetika.

Fraser mensimulasikan evolusi bilangan biner dari 15 bit sebagai string generasi dan menghitung presentase dari individu-individu yang terpilih oleh phenotype dengan generasi-generasi yang berurutan. Pada saat itu fraser tidak menyebutkan dalam laporannya bahwa algoritma pencarian dalam gejala alam akan berguna dalam sistem buatan, namun hasil dari penemuannya ternyata menyerupai optimasi fungsi.

Hal itulah yang memberikan inspirasi bagi John Holland dan murid-muridnya untuk mengaplikasikan proses genetika ini pada sistem buatan Holland yang menancapkan pondasi dalam karya tulisnya pada Teori Sistem Adaptif yaitu:

1. Concern Efficient Adaptive Systems (1962).
2. Information Prosessing and Prossesing Systems (1962).
3. Outline for a Logical Theory of Adaptive Systems (1962).

Tahun 1962-1965 John Holland memberikan kuliah mengenai masalah sistem adaptif di Universitas Michigan. Salah satu kuliahnya adalah *Theory of Adaptive Systems*. Dalam seminar-seminar John Holland dan murid-muridnya menyempurnakan detail dari algoritma genetika dan melakukan eksperimen dengan parameter-parameternya, terutama menciptakan rumus standar dari algoritma genetika. Selanjutnya John Holland menulis secara rinci mengenai teori Algoritma Genetika dalam buku karyanya, *Adaptation in Natural and Artificial System* yang dipublikasikan pada tahun 1975.[2]

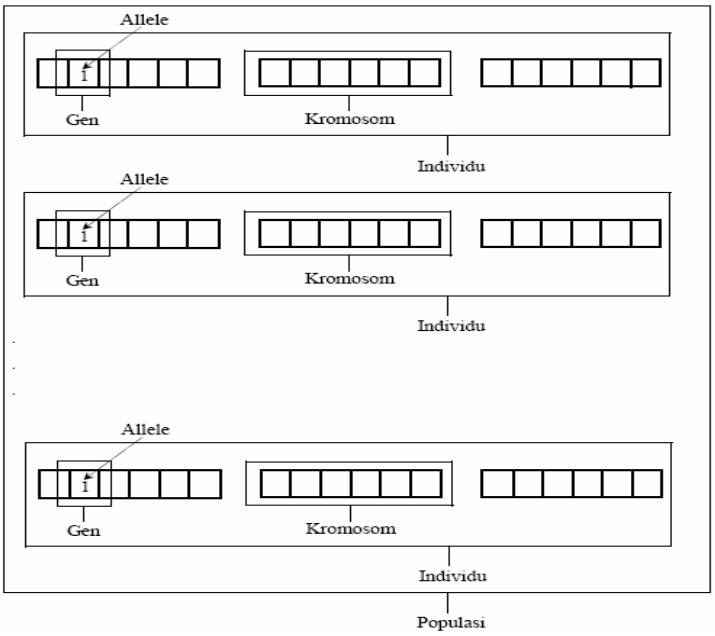
2.2.2 Pengertian Individu

Individu menyatakan salah satu solusi yang mungkin. Individu dapat dikatakan sama dengan kromosom, yaitu merupakan kumpulan gen. Beberapa definisi penting yang perlu diperhatikan dalam membangun penyelesaian masalah menggunakan Algoritma Genetika, yakni:

1. **Genotipe (Gen)**, sebuah nilai yang menyatakan satuan dasar yang membentuk suatu arti tertentu dalam satu kesatuan gen yang dinamakan kromosom. Dalam algoritma genetika, gen ini dapat berupa nilai biner, float, integer maupun karakter.
2. **Allele**, nilai dari gen.
3. **Kromosom**, gabungan gen yang membentuk nilai tertentu.

4. **Individu**, menyatakan suatu nilai atau keadaan yang menyatakan salah satu solusi yang mungkin dari permasalahan yang diangkat.
5. **Populasi**, merupakan sekumpulan individu yang akan diproses bersama dalam satu siklus proses evolusi.
6. **Generasi**, menyatakan satu siklus evolusi atau satu iterasi di dalam algoritma genetika.
7. **Phenotype**, string (kromosom) yang merupakan solusi akhir.

Gambar 2.2 berikut ini menjelaskan hubungan dari istilah-istilah yang didefinisikan di atas



Gambar 2.2 Ilustrasi definisi individu dalam algoritma genetika

2.2.3 Struktur Umum Algoritma Genetika

Sebuah solusi yang dibangkitkan dalam algoritma genetika disebut sebagai kromosom, sedangkan kumpulan kromosom yang berupa individu tersebut disebut sebagai populasi. Sebuah kromosom dibentuk dari komponen-komponen penyusun yang disebut sebagai gen dan nilainya dapat berupa bilangan numerik, biner, simbol ataupun karakter tergantung dari permasalahan yang ingin diselesaikan. Gen itu adalah komponen solusi dari suatu masalah yang akan diselesaikan. Kromosom-kromosom tersebut akan berevolusi secara berkelanjutan yang disebut dengan generasi. Pada setiap generasi kromosom-kromosom tersebut dievaluasi tingkat keberhasilan nilai

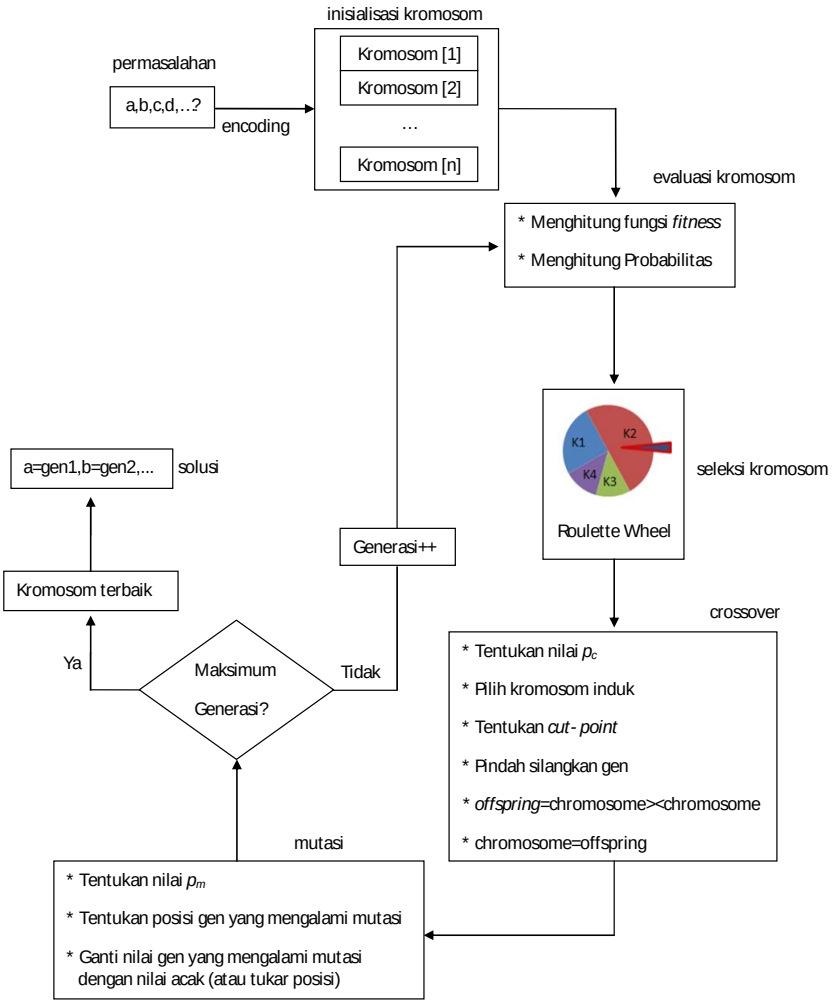
solusinya terhadap masalah yang ingin diselesaikan menggunakan ukuran kebugaran yang disebut dengan *fitness* (istilah didalam teknik optimasi, ini lebih dikenal sebagai fungsi tujuan *objective function*- atau fungsi biaya -*cost function*-) sebagai masalah yang akan dioptimalkan. *Fitness* Jika nilai *fitness* semakin besar, maka sistem yang dihasilkan akan semakin baik. Fungsi *fitness* ini ditentukan dengan menggunakan metode heuristik.

Untuk memilih kromosom yang tetap dipertahankan untuk generasi selanjutnya dilakukan proses yang disebut dengan seleksi (dengan proses tertentu, misalnya teknik *roulette wheel* untuk memilih pasangan dari induk yang akan dikawinkan). Proses seleksi kromosom menggunakan konsep aturan evolusi Darwin yang telah disebutkan sebelumnya yaitu kromosom yang mempunyai nilai *fitness* tinggi akan memiliki peluang lebih besar untuk terpilih lagi pada generasi selanjutnya.

Dalam perkawinan tersebut akan terjadi proses pertukaran gen antar individu yang kawin tersebut. Ada dua operator utama dalam proses tersebut, yakni perkawinan silang (*crossover*) dan mutasi. Dari hasil perkawinan tersebut dihasilkan keturunan berupa kromosom baru. Kromosom-kromosom baru itu disebut dengan *offspring* (anak) yang membawa beberapa sifat dari induknya. Jumlah kromosom dalam populasi yang mengalami pindah silang ditetukan oleh paramater yang disebut dengan probabilitas pindah silang (*crossover probability, Pc*).

Mekanisme perubahan susunan unsur penyusun mahkluk hidup akibat adanya faktor alam yang disebut dengan mutasi direpresentasikan sebagai proses berubahnya satu atau lebih nilai gen dalam kromosom dengan suatu nilai acak. Jumlah gen dalam populasi yang mengalami mutasi ditentukan oleh parameter yang dinamakan probabilitas mutasi (*mutation probability*, *Pmutasi*). Setelah beberapa generasi akan dihasilkan kromosom-kromosom yang nilai gen-gennya konvergen ke suatu nilai tertentu yang merupakan solusi optimum (atau yang lebih dikenal sebagai *acceptable optimum*) yang dihasilkan oleh algoritma genetika terhadap permasalahan yang ingin diselesaikan.

Struktur umum algoritma genetika di atas digambarkan pada gambar 2.3 berikut ini.



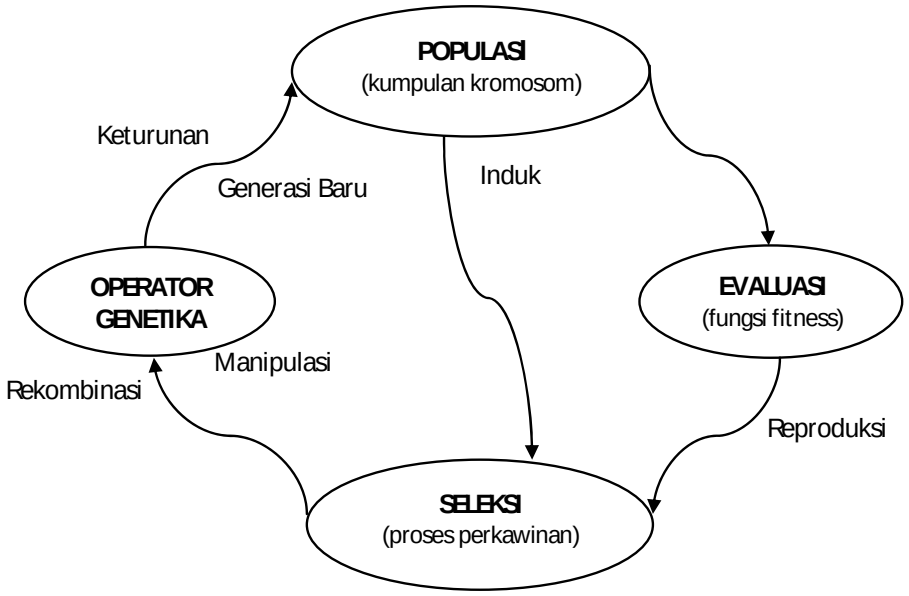
Gambar 2. 3 Diagram alir algoritma genetika

2.2.4 Komponen-komponen Algoritma Genetika

Secara umum sebuah penerapan Algoritma Genetika akan melalui siklus sederhana yang terdiri dari 4 langkah, yaitu:[2]

1. Membangun sebuah populasi yang terdiri dari beberapa string.
2. Evaluasi masing-masing string (*fitness value*).
3. Proses seleksi agar didapat string yang terbaik.
4. Manipulasi genetika untuk menciptakan populasi baru dari string.

Siklus 4 langkah yang dinspirasi dari proses biologi untuk algoritma genetika di atas dapat dilihat pada gambar di bawah ini.



Gambar 2.4 Siklus algoritma genetika

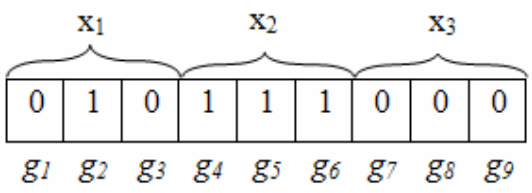
Setiap siklus yang dilalui memunculkan generasi baru yang dimungkinkan sebagai solusi bagi permasalahan yang ada. Pada dasarnya Algoritma Genetika memiliki tujuh komponen. Tetapi banyak metode yang bervariasi yang diusulkan pada masing-masing komponen tersebut. Masing-masing metode memiliki kelebihan dan kekurangan. Suatu metode yang bagus untuk penyelesaian masalah A belum tentu bagus untuk masalah B, atau bahkan tidak bisa digunakan untuk masalah C.

A. Skema Pengkodean

Pengkodean suatu kromosom adalah langkah pertama ketika menggunakan Algoritma Genetika untuk menyelesaikan suatu masalah. Pengkodean ini biasanya tergantung kepada masalah yang dihadapi. Pengkodean ini meliputi pengkodean terhadap gen yang terdapat dalam kromosom.

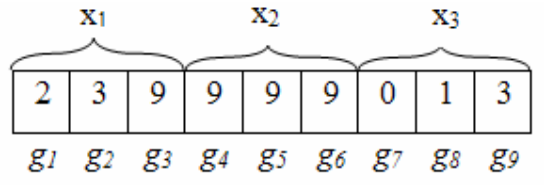
Terdapat tiga skema yang paling umum digunakan dalam sistem pengkodean, yaitu:

1. *Binary encoding*. Setiap gen hanya dapat bernilai 0 atau 1.



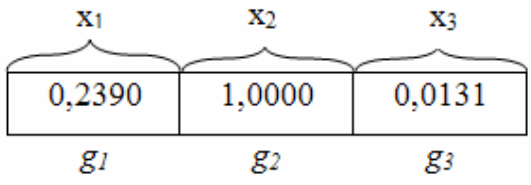
Gambar 2.5 Skema pengkodean *Binary encoding*

2. *Discrete decimal encoding*. Setiap gen dapat bernilai salah satu bilangan bulat dalam interval [0,9].



Gambar 2.6 Skema pengkodean *Discrete decimal encoding*

3. *Real-number encoding*. Pada skema ini, nilai gen berada dalam interval [0,R], di mana R adalah bilangan real positif dan biasanya R=1.



Gambar 2.7 Skema pengkodean *Real-number encoding*

4. *Permutation encoding*, setiap kromosom merupakan string dari sejumlah angka atau nomor yang merepresentasikan suatu posisi dalam suatu urutan.

Kromosom A 1 2 3 4 6 7 5 9 8

Kromosom B 2 4 6 7 8 9 1 5 3

Pada contoh gambar 2.5, 2.6 dan 2.7 skema pengkodean di atas terdapat tiga variabel, yaitu x_1, x_2, x_3 yang dikodekan ke dalam sebuah kromosom yang terdiri dari 9 gen yaitu g_1 sampai dengan g_9 (untuk *binary encoding* dan *discrete decimal encoding*) dimana setiap variabel dikodekan ke dalam 3 gen yaitu g_1 sampai dengan g_3, g_4 sampai dengan g_6, dan g_7 sampai dengan g_9. Sedangkan pada *real-number encoding* ketiga

variabel dikodekan ke dalam kromosom yang terdiri dari 3 gen, dimana masing-masing variabel dikodekan ke dalam 1 gen, variabel x_1 dikodekan ke g_1, variabel x_2 dikodekan ke g_2, dan variabel x_3 dikodekan g_3.

B. Nilai Fitness

Suatu individu dievaluasi berdasarkan suatu fungsi tertentu sebagai ukuran performansinya. Di dalam evolusi alam, individu yang memiliki nilai *fitness* tinggi yang akan bertahan hidup. Sedangkan individu yang bernilai *fitness* rendah akan mati. Dalam masalah optimasi, jika masalah yang dicari adalah memaksimalkan sebuah fungsi *h* (dikenal sebagai masalah maksimasi) maka nilai *fitness* yang digunakan adalah nilai dari fungsi *h* tersebut, yakni *f = h* (dimana *f* adalah nilai *fitness*). Tetapi jika masalahnya adalah meminimalkan fungsi *h* (masalah minimasi), maka fungsi *h* tidak dapat digunakan secara langsung. Hal ini disebabkan adanya aturan bahwa individu yang memiliki *fitness* tertinggi lebih mampu bertahan hidup pada generasi berikutnya. Oleh karena itu nilai *fitness* yang dapat digunakan adalah *f = 1/h*, yang artinya semakin kecil nilai *h*, semkin besar nilai *f*. Pada masalah-masalah tertentu *h* dapat bernilai 0, yang mengakibatkan *f* dapat bernilai tak hingga. Untuk mengatasinya, *h* perlu ditambah sebuah bilangan yang dianggap sangat kecil sehingga nilai *fitness*-nya menjadi :

$$f = \frac{1}{(a+h)} \quad \text{..... (2.6)}$$

dimana *a* adalah bilangan yang dianggap sangat kecil dan bervariasi sesuai dengan masalah yang akan diselesaikan.

C. Seleksi Induk

Pemilihan dua buah kromosom sebagai induk, yang akan dipindahsilangkan, biasanya dilakukan secara proporsional sesuai dengan nilai *fitness*-nya.

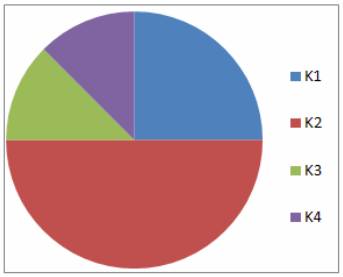
Beberapa metode yang biasa digunakan :

1. Roulette-wheel

   Sesuai dengan namanya, metode ini menirukan permainan *roulette-wheel* dimana masing-masing kromosom menempati potongan lingkaran pada roda *roulette* secara proporsional sesuai dengan nilai *fitness*-nya. Kromosom yang memiliki nilai *fitness* lebih besar menempati potongan lingkaran yang lebih besar dibandingkan dengan kromosom yang bernilai fitness rendah.

Gambar di bawah ini mengilustrasikan sebuah contoh penggunaan *roulette wheel.*

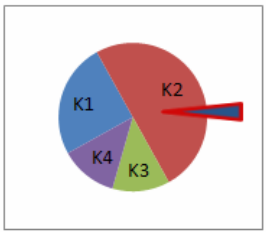
Kromosom	Nilai *Fitness*
K1	1
K2	2
K3	0.5
K4	0.5
Jumlah	4



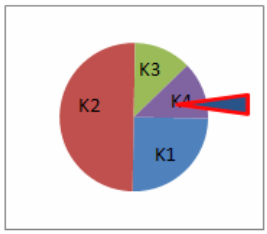
Gambar 2.8 Contoh penggunaan metode *Roulette-wheel*

Metode *roullette-wheel* sangat mudah diimplementasikan dalam pemprograman. Pertama, dibuat interval nilai kumulatif (dalam interval [0,1]) dari nilai *fitness* masing-masing kromosom dibagi total nilai *fitness* dari semua kromosom. Sebuah kromosom akan terpilih jika bilangan random yang dibangkitkan berada pada interval nilai kumulatifnya. Pada Gambar 2.8 di atas, K1 menempati interval nilai kumulatif [0;0,25], K2 berada dalam interval [0,25;0,75], K3 dalam interval [0,75;0,875] dan K4 dalam interval [0,875;1].

Misalkan, jika bilangan random yang dibangkitkan adalah 0,6 maka kromosom K2 terpilih sebagai orang tua. Tetapi jika bilangan random yang dibangkitkan 0,99 maka kromosom K4 yang terpilih. Seperti diilustrasikan pada gambar berikut ini:



Bilangan random=0,6



Bilangan random=0,99

Gambar 2.9 Ilustrasi prinsip kerja metode *Roulette-wheel*

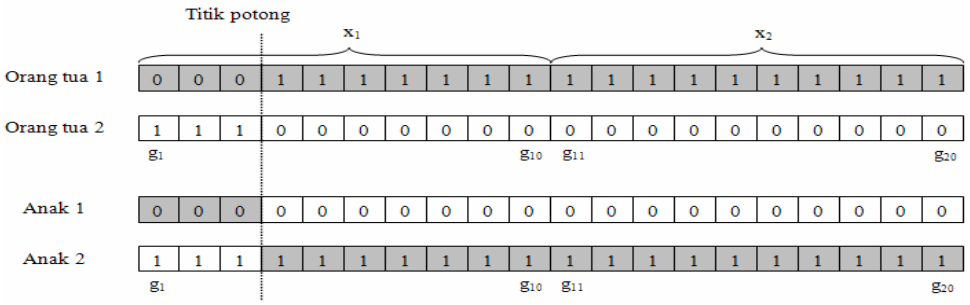
2. Kompetisi (Tournament)

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.

Metode *tournament* yang lebih rumit adalah dengan mengambil *m* kromosom secara random. Kemudian kromosom bernilai fitness tertinggi dipilih menjadi orang tua pertama, apabila bilangan random yang dibangkitkan kurang dari suatu nilai batas yang ditentukan *p* dalam interval [0,1], pemilihan orang tua akan dilakukan secara random dari *m* - 1 kromosom yang ada jika bilangan yang dibangkitkan lebih dari atau sama dengan *p*. Pada *tournament selection*, variabel *m* adalah jumlah kromosom dan *p* adalah peluang kompetisi (*tournament probability)*. Biasanya *m* diset sebagai suatu nilai yang sangat kecil, misal 4 atau 5. Sedangkan *p* biasanya diset sekitar 0,75.

D. Pindah Silang (Crossover)

Salah satu komponen paling penting dalam Algoritma Genetika adalah *crossover* atau pindah silang. Sebuah kromosom yang mengarah pada solusi yang bagus bisa diperoleh dari proses memindah-silangkan dua buah kromosom. Contoh pindah silang :



Gambar 2.10 Contoh proses pindah silang

Pada gambar contoh pindah silang di atas apabila solusi yang dicari adalah $x_1 = 0$ dan $x_2 = 0$, maka kromosom Anak 1 memiliki nilai *fitness* tinggi dan menuju pada solusi yang dicari.

Pindah silang bisa juga berakibat buruk jika ukuran populasinya sangat kecil. Dalam suatu populasi yang sangat kecil, suatu kromosom dengan gen-gen yang mengarah ke solusi akan sangat cepat menyebar ke kromosom-kromosom lainnya. Untuk mengatasi masalah ini digunakan suatu aturan bahwa pindah silang hanya bisa dilakukan dengan suatu probabilitas tertentu *Pc*. Artinya, pindah silang bisa dilakukan hanya jika suatu bilangan random [0,1] yang dibangkitkan kurang dari *Pc* yang ditentukan. Pada umumnya *Pc* diset mendekati 1, misalnya 0,8.

Pindah silang bisa dilakukan dalam beberapa cara berbeda, yang paling sederhana adalah pindah silang satu titik potong (*one-point crossover*). Suatu titik potong dipilih secara random, kemudian bagian pertama dari orang tua 1 digabung dengan bagian kedua dari orang tua 2. Untuk kromosom yang sangat panjang, misalkan 1000 gen, mungkin saja diperlukan beberapa titik potong. Pindah silang lebih dari satu titik potong disebut *n-point crossover*, dimana *n* titik potong dipilih secara random dan bagian-bagian kromosom dipilih dengan probabilitas 0,5 dari salah satu orang tuanya.[1]

E. Mutasi

Prosedur mutasi sangatlah sederhana. Untuk semua gen yang ada, jika bilangan random yang dibangkitkan kurang dari probabilitas mutasi *Pmutasi* yang ditentukan maka ubah gen tersebut menjadi nilai kebalikannya (dalam binary encoding, 0 diubah 1, dan 1 diubah 0). Dalam *swapping mutation* dilakukan pertukaran tempat node. Biasanya *Pmutasi* diset sebagai 1/*n*, dimana *n* adalah jumlah gen dalam kromosom. Dengan *Pmutasi* sebesar ini berarti mutasi terjadi pada sekitar satu gen saja. Pada algoritma genetika sederhana, nilai *Pmutasi* tetap selama evolusi.

Beberapa skema mutasi yang biasa digunakan :

1. Mutasi biner, contoh : 1**1**001001 => 1**0**001001
2. *Swapping mutation*, contoh : (1 **2** 3 4 5 6 **7** 9 8) => (1 **7** 3 4 5 6 **2** 9 8)

F. Elitisme

Karena seleksi dilakukan secara random, maka tidak ada jaminan bahwa suatu individu bernilai *fitness* tertinggi akan selalu terpilih. Kalaupun individu bernilai *fitness* tertinggi terpilih, mungkin saja individu tersebut akan rusak (nilai *fitness*-nya menurun)

karena proses pindah silang. Untuk menjaga agar individu bernilai *fitness* tertinggi tersebut tidak hilang selama evolusi, maka perlu dibuat satu atau beberapa *copy*-nya. Prosedur ini dikenal sebagai *elitisme*.

G. Pergantian Populasi

Pada Agoritma Genetika dikenal skema pergantian populasi dalam suatu generasi yaitu *generational replacement*, yang berarti semua individu (misal N individu dalam suatu populasi) dari suatu generasi digantikan sekaligus oleh N individu baru hasil pindah silang dan mutasi. Skema penggantian ini tidak realistis dari sudut pandang biologi. Di dunia nyata, individu-individu dari generasi berbeda bisa berada pada waktu yang bersamaan. Fakta lainnya adalah individu-individu muncul dan hilang secara konstan, tidak pada generasi tertentu. Secara umum skema pergantian populasi dapat dirumuskan berdasarkan suatu ukuran yang disebut *generational gap* G. Ukuran ini menunjukkan persentase populasi yang digantikan dalam setiap generasi. Pada skema *generational replacement*, $G = 1$.

Skema pergantian yang paling ekstrim adalah hanya mengganti satu individu dalam setiap generasi, yaitu $G = 1/N$, dimana N adalah jumlah individu dalam populasi. Skema penggantian ini disebut sebagai *Steady-state reproduction*. Pada skema tersebut, G biasanya sama dengan $1/N$ atau $2/N$. Dalam setiap generasi, sejumlah NG individu harus dihapus untuk menjaga ukuran populasi tetap N. Terdapat beberapa prosedur penghapusan individu, yaitu penghapusan individu yang bernilai fitness paling rendah atau penghapusan individu yang paling tua. Penghapusan bisa berlaku hanya pada individu orang tua saja atau bisa juga berlaku pada semua individu dalam populasi.[1]

2.3 Travelling Salesman Problem dalam Algoritma Genetika

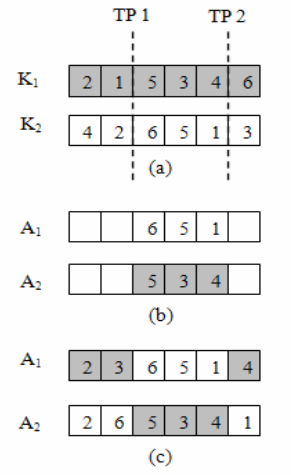
TSP dapat dirumuskan sebagai berikut: terdapat sekumpulan N node dengan posisi-posisi koordinatnya $\{x,y\}$, $i = 1, 2, 3, \ldots, N$. Algoritma Genetika dapat digunakan untuk menyelesaikan masalah TSP dengan cara sebagai berikut:

1. Suatu solusi direpresentasikan ke dalam suatu kromosom yang berisi nomor urut dari semua node yang ada. Masing-masing nomor urut node hanya boleh muncul satu kali di dalam kromosom sehingga satu kromosom merepresentasikan satu rute atau satu solusi yang valid. Representasi ini disebut sebagai *permutation encoding*, dimana suatu kromosom merepresentasikan suatu permutasi dari nomor urut kota $1, 2, 3, \ldots, N$.

2. Kromosom-kromosom tersebut diperiksa nilai *fitness*-nya, kromosom yang nilai *fitness*-nya paling tinggi akan dipilih untuk terus hidup pada generasi berikutnya dan berpeluang melakukan *crossover* untuk menghasilkan kromosom atau individu baru yang diharapkan mempunyai nilai *fitness* yang lebih baik. Dengan adanya mutasi diharapkan dapat memperbaiki kromosom yang sudah ada.
3. Individu-individu pada generasi-generasi berikutnya diharapkan akan memiliki nilai *fitness* yang lebih baik dan mengarah pada suatu solusi yang diharapkan. Solusi yang diambil adalah solusi pada individu atau kromosom yang paling besar nilai *fitness*-nya.

Dalam TSP, pindah silang dapat diimplementasikan dengan skema *order crossover*. Pada skema ini satu bagian kromosom ditukarkan dengan tetap menjaga urutan node (gen) yang bukan bagian dari kromosom tersebut.

Berikut ini adalah ilustrasi skema *order crossover*:



Gambar 2.11 Pindah silang menggunakan *order crossover*
(a) Posisi titik potong 1 dan 2 (TP1 dan TP2) pada kromosom induk (K1 dan K2)
(b) Gen pada kromosom Anak 1 dan 2 (A1 dan A2)
(c) Posisi gen Anak 1 dan 2 (A1 dan A2)

Mula-mula dua buah titik potong TP1 dan TP2, dibangkitkan secara random untuk memotong dua buah kromosom induk (orang tua), K1 dan K2. Kemudian dua kromosom anak, A1 dan A2 mendapat gen-gen dari bagian kromosom K1 dan K2 secara menyilang. Kromosom A1 mendapatkan {6, 5, 1} dan A2 mendapatkan {5, 3, 4}.

Posisi gen yang masih kosong pada A1 diisi dengan gen-gen dari K1, secara berurutan dari gen 1 sampai gen 6, yang belum ada pada A1. Hal yang sama juga dilakukan untuk kromosom A2.

Pada TSP, operator mutasi biasanya diimplementasikan dengan menurunkan gen termutasi dengan gen lain yang terpilih secara random (acak). Misalnya, kromosom {2, 3, 4, 5, 6, 1} dapat termutasi menjadi kromosom {2, 6, 4, 5, 3, 1}. Dalam hal ini gen 3 dan 6 saling ditukar. Skema mutasi ini dikenal dengan *swapping mutation*.