BAB II
KAJIAN TEORI

1. Teori graf

1. Definisi Graf

G membentuk suatu graf jika terdapat pasangan himpunan $(V(G), E(G))$, dimana $V(G)$ (simpul pada graf G) tidak kosong dan $E(G)$ (rusuk pada graf G). Jika v dan w adalah sepasang simpul yang berbeda di G, vw melambangkan rusuk di G dan jika $e = vw$ adalah rusuk di G maka:

a. v dan w berikatan *(adjacent)* di G
b. rusuk e hadir *(joining)* simpul v dan w di G
c. v dan w adalah simpul ujung rusuk di e
d. rusuk e hadir (*incident*) di simpul v dan w atau sebaliknya dikatakan simpul v dan w hadir pada rusuk e.

Menutut Rosen berdasarkan ada tidaknya bobot, graf dikelompokkan menjadi dua jenis yaitu graf berbobot dan graf tak-berbobot.

a. Graf Berbobot

Suatu graf dikatakan sebagai graf berbobot jika setiap rusuknya mempunyai nilai atau bobot tertentu. Bobot pada graf biasanya dinotasikan dengan w_{ij} dengan i dan j sebagai simpul yang terhubung dengan rusuk yang memiliki bobot w.

b. Graf Tak-Berbobot

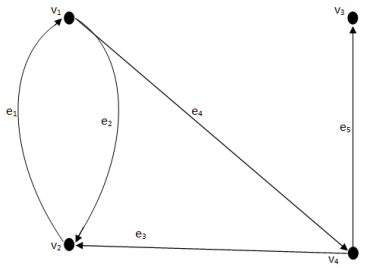
Suatu graf dikatakan sebagai graf tidak berbobot jika setiap rusuknya tidak mempunyai nilai atau bobot tertentu.

Berdasarkan orientasi arah, menurut Rosen graf dikelompokkan menjadi dua jenis yaitu graf berarah dan graf tak-berarah.

a. Graf Berarah *(Directed Graph)*

Graf berarah adalah graf yang rusuknya mempunyai orientasi arah.

Contoh 2.1



Gambar 2.1 Graf D

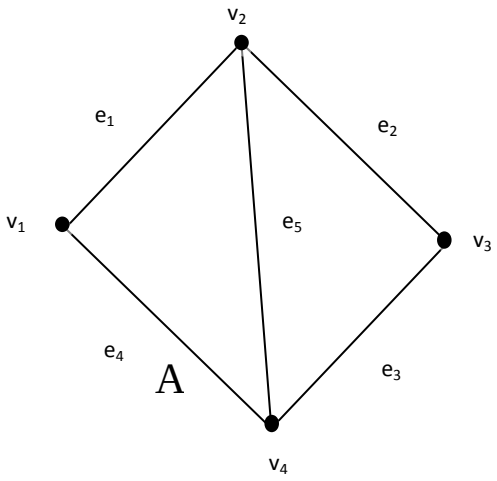
Graf D pada gambar 2.5 memiliki $V(D)=(v_1,v_2,v_3,v_4)$, $E(D)=(e_1,e_2,e_3,e_4,e_5)$, sedangkan $e_1=(v_2,v_1)$, $e_2=(v_1,v_2)$, $e_3=(v_4,v_2)$, $e_4=(v_1,v_4)$, $e_5=(v_4,v_3)$ Graf D pada gambar 2.6 menunjukkan rusuk e_1 tidak sama dengan e_2.

b. Graf Tak-Berarah *(Undirected Graph)*

Definisi 2.11(Rosen, 2003: 47)

Graf tak berarah adalah graf yang rusuknya tidak mempunyai orientasi arah.

Contoh 2.2



Gambar 2.2 Graf A

Graf A dari gambar 2.1 merupakan contoh graf tak-berarah.

2. Keterhubungan Graf

a. Pengertian Dasar Pada Graf

a) Jalan (*walk*)

Misal G adalah graf. Sebuah pengertian jalan *(walk*) di G adalah sebuah barisan berhingga (tak kosong) yang suku-sukunya bergantian simpul dan rusuk, sedemikian hingga v_{i-1}dan v_i adalah simpul-simpul dari rusuk e_i , dan dinotasikan $W = (v_0, e_1, v_1, e_2, \ldots, e_k, v_k)$ untuk $1 \leq i \leq k$.

b) Jejak *(trail*)

Jejak *(trail*) pada graf G adalah jalan tanpa rusuk berulang di graf G. Misal $W = (v_0, e_1, v_1, e_2, \ldots, e_k, v_k)$ adalah sebuah jalan di graf G, maka W disebut jejak jika semua rusuk $e_1, e_2, e_3, \ldots, e_k$ dalam jalan W berbeda.

c) Lintasan (*path*)

Lintasan (*path*) adalah sebuah *trail* tanpa simpul berulang.

d) Sikel (*cycle*)

Sikel (*cycle*) adalah sebuah jejak tertutup (*closed trail*) yang simpul awal dan akhir merupakan simpul yang sama.

b. Graf terhubung

Sebuah graf G disebut terhubung jika untuk setiap dua simpul u dan v di G terdapat lintasan di G yang menghubungkan kedua simpul tersebut, sebaliknya graf G disebut graf tidak terhubung jika untuk setiap dua simpul u dan v di G tidak terdapat lintasan di G yang menghubungkan kedua simpul tersebut.

2. *Vehicle Routing Problem* (VRP)

Vehicle Routing Problem (VRP) didefinisikan sebagai masalah penentuan rute optimal untuk pendistribusian barang/jasa ke pelanggan-pelanggan dengan lokasi yang berbeda dengan permintaan yang sudah diketahui, dari satu atau lebih depot yang memenuhi beberapa kendala (Yeun dkk, 2008). Masalah ini merupakan generalisasi dari *m-Traveling Salesman Problem* (*m*-TSP) dengan diberikan himpunan *N* kota dan seorang *salesman* yang ingin menemukan jalur terpendek untuk mengunjungi setiap kota tepat satu kali dan selesai di kota asal (Ho, Lim, & Oon, 2001). Pada m-TSP terdapat m salesman yang mengunjungi N kota tepat satu kali, sedangkan pada VRP kota-kota pada m-TSP

merupakan pelanggan dan salesman merupakan kendaraan, dimana tiap kendaraan memiliki kapasitas tertentu sehingga total permintaan dari satu rute tidak boleh melebihi kapasitas yang dimiliki salesman. VRP dengan kendala kapasitas *disebut Capacitated Vehicle Routing Problem* (CVRP).

CVRP merupakan salah satu contoh permasalahan pada VRP, contoh permasalahan VRP selain CVRP adalah (Solomon, 1987)

a. *Vehicle Routing Problem with Pickup and Delivery* (VRPPD) merupakan VRP dengan permintaan yang terdiri dari penjemput dan pengantaran.
b. *Dynamic Vehicle Routing Problem* (DVRP) merupakan VRP yang terdapat penambahan pelanggan baru saat kendaraan sedang melayani pelanggan.
c. *Vehicle Routing Problem with Time Windows* (VRPTW) merupakan CVRP dengan penambahan kendala waktu (*time windows*) pada masing-masing pelanggan dan depot
d. *Split delivery* VRP (SDVRP), yaitu pelanggan dilayani dengan kendaraan berbeda
e. *Stochastic* VRP (SVRP), yaitu munculnya ‘random values’ (seperti jumlah pelanggan, jumlah permintaan, waktu pelayanan atau waktu perjalanan)
f. *Periodic* VRP, yaitu pengantar hanya dilakukan dihari tertentu

3. *Capacitated Vehicle Routing Problem* (CVRP)

Capacitated vehicle routing problem (CVRP) merupakan salah satu permasalahan pada VRP. Kendala pada kasus CVRP yaitu terdapat kapasitas pada setiap kendaraan. CVRP bertujuan untuk meminimumkan total jarak tempuh perjalanan kendaraan dan meminimumkan banyaknya kendaraan yang digunakan dalam mendistibusikan barang dari depot ke konsumen.

Masalah utama dalam masalah CVRP adalah bagaimana menentukan rute untuk K kendaraan tersebut sedemikian sehingga setiap pelanggan terlayani oleh tepat satu kendaraan, permintaan terpenuhi, muatan sepanjang rute tidak melampaui kapasitas W, panjang rute dari depot keliling kembali ke depot lagi tidak melampaui T dan akhirnya jumlah total panjang rute seluruh K kendaraan minimum (*Sarwadi, 1995:2*).

Menurut Tonci Caric dan Hrvoje Gold CVRP sebagai suatu graf berarah $G = (V, A)$ dengan $V = \{v_0, v_1, v_2, \ldots, v_n, v_{n+1}\}$ adalah himpunan simpul (verteks), v_0 menyatakan depot dengan v_{n+1} merupakan depot semu dari v_0 yaitu tempat kendaraan memulai dan mengakhiri rute perjalanan. Sedangkan $A = \{v_i, v_j : v_i, v_j \in V, i \neq j\}$ adalah himpunan sisi berarah (arc) yang merupakan himpunan sisi yang menghubungkan antar simpul. Setiap simpul $v_i \in V$ memiliki permintaan(demand) sebesar d_i dengan d_j adalah integer positif. Himpunan $K = \{k_1, k_2, \ldots, k_n\}$ merupakan kumpulan kendaraan yang homogen dengan kapasitas yang identik yaitu Q, sehingga panjang setiap rute dibatasi oleh kapasitas

kendaraan. Setiap verteks (v_i, v_j) memiliki jarak tempuh C_{ij} yaitu jarak dari simpul i ke simpul j. Jarak perjalanan ini diasumsikan simetrik yaitu $C_{ij} = C_{ji}$ dan $C_{ii} = 0$.

$$\min \qquad z = \sum_{i=0}^{N} \sum_{j=0}^{N} (C_{ij} \sum_{k=1}^{K} X_{ijk})$$

Tujuan penyelesaian CVRP yaitu meminimumkan jumlah jarak rute perjalanan kendaraan dengan kendala-kendala sebagai berikut (Sri Nurhayati, 2013: 4-5)

1. Setiap simpul hanya dikunjungi tepat satu kali oleh kendaraan
2. Total jumlah permintaan konsumen dalam satu rute tidak melebihi kapasitas kendaraan yang melayani rute tersebut.
3. Setiap rute perjalanan berawal dari depot
4. Setiap rute perjalanan berakhir di depot
5. Kekontinuan rute, artinya kendaraan yang mengunjungi suatu simpul, setelah selesai melayani akan meninggalkan simpul tersebut
6. Tidak terdapat sub rute pada setiap rute yang terbentuk
7. Variabel keputusan X_{ijk} merupakan integer biner

Dari permasalahan CVRP maka di formulasikan dalam bentuk model matematika pada tabel 2.1

Tabel 2.1 Model matematika CVRP

Fungsi tujuan	min $z = \sum_{i=1}^{n} \sum_{j=1}^{n} (C_{ij} \sum_{k=1}^{m} X_{ijk})$	
Kendala tujuan	$\sum_{i=1}^{n} \sum_{k=1}^{m} X_{i1k} = 1$ $\vdots$ $\sum_{i=1}^{n} \sum_{k=1}^{m} X_{ink} = 1$	
	$\sum_{i=1}^{n} \sum_{k=1}^{m} X_{1jk} = 1$ $\vdots$ $\sum_{i=1}^{n} \sum_{k=1}^{m} X_{njk} = 1$	
	$\sum_{i=1}^{n} (d_i \sum_{j=1}^{n} X_{ijk}) \leq W$	$k = 1, 2, \ldots, m$
	$\sum_{j=1}^{N} X_{0jk} = 1$	$k = 1, 2, \ldots, m$
	$\sum_{i=1}^{N} X_{i0k} = 1$	$k = 1, 2, \ldots, m$
	$\sum_{i=0}^{n} X_{ilk} - \sum_{i=0}^{n} X_{ilk} = 0$	$k = 1, 2, \ldots, m$ $l = 0, 1, \ldots, n$
	$X_{ijk} \in \{0,1\}$	$\forall i, j \in V$ $i \neq j$ $k = 1, 2, \ldots, m$

Keterangan:
K $= \{k_1, k_2, \ldots, k_m\}$ kendaraan yang digunakan
V = himpunan simpul
A = himpunan rusuk berarah (arc), $\{ (v_i, v_j) : v_i, v_j \in V \; i \neq j \}$
C_{ij} = jarak antara simpul v_i ke simpul v_j
d_i = jumlah permintaan pada simpul v_i
W = kapasitas masing-masing kendaraan

4. Algoritma Genetika

1. Definisi Algoritma Genetika

Algoritma genetika (AG) didasarkan pada prinsip seleksi alam yaitu "siapa yang kuat, dia yang bertahan". AG pertama kali ditemukan oleh John Holland pada tahun 1960. Bersama murid dan teman-temannya, John Holand mepublikasikan AG dalam buku yang berjudul *Adaption of Natural and Artificial Systems* pada tahun 1975 (Coley, 1999). AG merupakan algoritma optimisasi

yang terinspirasi oleh gen dan seleksi alam. Algoritma ini mengodekan solusi-solusi yang mungkin ke dalam struktur data dalam bentuk kromosom-kromosom dan mengaplikasikan operasi rekombinasi genetik ke struktur data tersebut (Whitley, 2002).

Hal-hal yang terdapat dalam algoritma genetika adalah sebagai berikut (Satriyanto, 2009).

a. Gen (*Genotype*) adalah sebuah nilai yang menyatakan satuan dasar yang membentuk suatu arti tertentu dalam satu kesatuan gen yang dinamakan kromosom.
b. *Allele* yaitu nilai dari sebuah gen, dapat berupa bilangan biner, float, integer, karakter dan kombinatorial.
c. Kromosom adalah gabungan gen – gen yang membentuk nilai tertentu.
d. Individu merupakan suatu nilai atau keadaan yang menyatakan salah satu solusi yang mungkin dari permasalahan yang diangkat.
e. Populasi merupakan sekumpulan individu yang akan diproses bersama dalam satu siklus proses evolusi. Populasi terdiri dari sekumpulan kromosom.
f. Induk, adalah kromosom yang akan dikenai operasi genetik (*crossover*)
g. *Crossover* merupakan operasi genetik yang mewakili proses perkembangbiakan antar individu.

h. *Offspring* adalah kromosom yang merupakan hasil dari operasi genetik (*crossover*) dikenal keturunan atau sebagai anak.

i. Mutasi merupakan operasi genetik yang mewakili proses mutasi dalam perjalanan hidup individu. Mutasi berperan menghasilkan perubahan acak dalam populasi, yang berguna untuk menambah variasi dari kromosom – kromosom dalam sebuah populasi.

j. Proses Seleksi merupakan proses yang mewakili proses seleksi alam (*natural selection*) dari teori Darwin. Proses ini dilakukan untuk menentukan induk dari operasi genetik (*crossover*) yang akan dilakukan untuk menghasilkan keturunan (*offspring*).

k. Nilai *fitness* merupakan penilaian yang menentukan bagus tidaknya sebuah kromosom.

l. Fungsi Evaluasi adalah fungsi yang digunakan untuk menentukan nilai *fitness*. Fungsi evaluasi ini merupakan sekumpulan kriteria-kriteria tertentu dari permasalahan yang ingin diselesaikan.

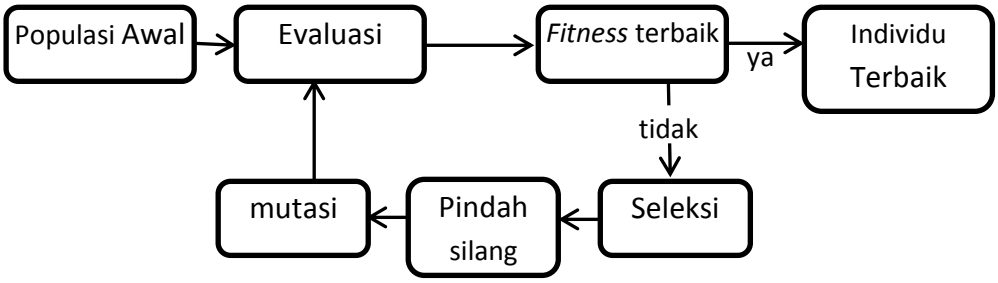
m. Generasi merupakan satuan dari populasi setelah mengalami operasi-operasi genetika, berkembang biak, dan menghasilkan keturunan. Pada akhir dari setiap generasi, untuk menjaga agar jumlah kromosom dalam populasi tetap konstan, kromosom–kromosom yang mempunyai Nilai

fitness yang rendah dan memiliki peringkat dibawah nilai minimal akan dihapus dari populasi.

Secara umum, proses algoritma genetika adalah sebagai berikut (Kusumadewi, 2003: 92).

1. Membangkitkan populasi awal secara acak.
2. Membentuk generasi baru dengan menggunakan operasi seleksi, operasi *crossover* dan operasi mutasi secara berulang-ulang sehingga diperoleh kromosom yang cukup untuk membentuk generasi baru sebagai representasi dari solusi baru.
3. Mengevaluasi setiap populasi dengan menghitung nilai *fitness* setiap kromosom hingga terpenuhi kriteria berhenti. Bila kriteria berhenti belum terpenuhi, maka akan dibentuk lagi generasi baru dengan mengulangi langkah 2. Kriteria berhenti yang digunakan adalah sebagai berikut.
   a. Berhenti pada generasi tertentu.
   b. Berhenti setelah dalam beberapa generasi berturut-berturut didapatkan nilai *fitness* tertinggi yang tidak berubah (*konvergen*).
   c. Berhenti bila dalam n generasi berikutnya tidak didapatkan nilai *fitness* yang lebih optimal.

Proses algoritma genetika di atas diilustrasikan pada gambar 2.3 berikut.



Gambar 2.3 *Flow chart* algoritma genetika

2. **Komponen-Komponen Utama dalam Algoritma Genetika**

Komponen-komponen utama dalam menggunakan algoritma genetika sebagai berikut.

a. Penyandian Gen (Pengkodean)

Teknik penyandian adalah proses penyandian gen dari kromosom. Gen merupakan bagian dari kromosom, satu gen biasanya akan mewakili satu variabel. Gen dapat direpesentasikan dalam bentuk bit, bilangan real, daftar aturan, elemen permutasi, elemen program atau representasi lainnya yang dapat diimplementasikan dalam operator genetika (Satriyanto, 2009). Terdapat beberapa teknik pengkodean dalam algoritma genetika diantaranya pengkodean biner, pengkodean permutasi, pengkodean nilai dan pengkodean pohon (Anwar dan Yuliani, 2005).

Pada penelitian ini, representasi gen menggunakan teknik pengkodean permutasi. Dalam pengkodean ini, tiap gen dalam kromosom merepresentasikan suatu urutan (Anwar dan Yuliani, 2005).

Contoh 2.3 kromosom 1 = 2 3 4 5 1 6 7

Keterangan: kromosom 1 berisi urutan secara acak gen kesatu sampai ke tujuh. Gen direpresentasikan dengan sebuah bilangan dan bilangan-bilangan tersebut representasi dari masing-masing kota.

b. Membangkitkan Populasi Awal (*Spanning*)

Membangkitkan populasi awal adalah membangkitkan sejumlah individu secara acak atau melalui prosedur tertentu. Ukuran populasi tergantung pada masalah yang akan dipecahkan dan jenis operator genetika yang akan diimplementasikan. Setelah ukuran populasi ditentukan, kemudian harus dilakukan inisialisasi terhadap kromosom yang terdapat pada populasi tersebut. Inisialisasi kromosom dilakukan secara acak, namun demikian harus tetap memperhatikan domain solusi dan kendala permasalahan yang ada (Kusumadewi, 2003: 102).

Terdapat berbagai teknik dalam pembangkitan populasi awal ini yaitu *random generator*, pendekatan tertentu dan permutasi gen. Pada penelitian ini, pembangkitan populasi awal dengan menggunakan *random generator*. *Random generator* melibatkan pembangkitan bilangan random dalam interval (0,1) untuk nilai setiap gen sesuai dengan representasi kromosom yang digunakan.

c. Evaluasi Nilai *Fitness (Fitness Value)*

Evaluasi nilai *fitness* berfungsi untuk mengukur kualitas dari sebuah solusi dan memungkinkan tiap solusi untuk dibandingkan (Michalewicz, 1996: 72). Suatu individu dievaluasi berdasarkan suatu fungsi tertentu sebagai ukuran baik tidaknya individu tersebut. Di dalam evolusi alam, individu yang bernilai *fitness* tinggi yang akan bertahan hidup, sedangkan individu yang bernilai *fitness* rendah akan mati (D.E.Goldberg, 1989). Pada masalah optimasi, fungsi *fitness* yang yang digunakan adalah

$$f = \frac{1}{x}, \qquad (2.1)$$

dengan x merupakan nilai dari individu, yang artinya semakin kecil nilai x, maka semakin besar nilai *fitness*nya. Tetapi hal ini akan menjadi masalah jika x bernilai 0, yang mengakibatkan *f* bisa bernilai tak hingga jika x=0. Untuk mengatasinya, x perlu ditambah sebuah bilangan sangat kecil sehingga nilai *fitness*nya menjadi

$$f = \frac{1}{(x+a)}, \qquad (2.2)$$

dengan a adalah bilangan yang dianggap sangat kecil.

d. Seleksi *(Selection)*

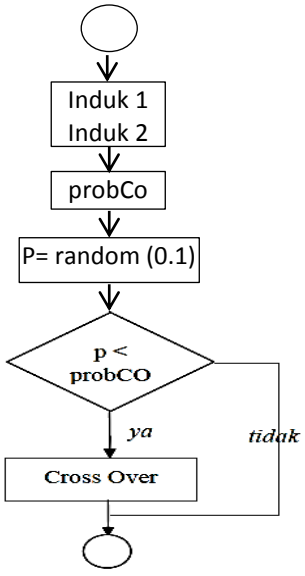
Seleksi merupakan pemilihan dua buah kromosom untuk dijadikan sebagai induk yang dilakukan secara proporsional sesuai dengan dengan nilai *fitness*-nya (Michalewicz, 1996: 75). Masing-masing individu yang diseleksi akan diberikan probabilitas reproduksi tergantung dari nilai objektif dirinya sendiri terhadap nilai objektif dari semua individu dalam seleksi tersebut. Nilai *fitness* inilah yang nantinya akan digunakan pada tahap seleksi berikutnya.

Terdapat beberapa metode seleksi menurut Kusumadewi (2003:105), yaitu *rank-based fitness assignment*, *roulette wheel selection*, s*tochastic universal sampling*, seleksi lokal (l*ocal selection*), seleksi dengan pemotongan (*truncation selection*) dan seleksi dengan turnamen (*tournament selection*).

e. *Crossover* (Pindah Silang)

Pindah Silang (*crossover*) adalah operator dari algoritma genetika yang melibatkan dua induk untuk membentuk kromosom baru. Pindah silang menghasilkan keturunan baru dalam ruang pencarian yang siap diuji. Operasi ini tidak selalu dilakukan pada setiap individu yang ada. Individu dipilih secara acak untuk dilakukan *crossover* dengan Pc (*Probabilitas Crossover*) antara 0,6 s/d 0,95. Jika pindah silang tidak diakukan, maka nilai dari induk akan diturunkan kepada keturunan (Michalewicz, 1996: 78).

Prinsip dari pindah silang ini adalah melakukan operasi pertukaran pada gen yang bersesuaian dari induk untuk mengasilkan individu baru. Proses *crossover* dilakukan pada setiap individu dengan probabilitas crossover yang ditentukan. Secara skematis proses *cross-over* seperti Gambar 2.4



Gambar 2.4 Sistematika proses cross-over

Dari gambar 2.4, jika bilangan p yang dibangkitkan secara acak kurang dari probabilitas *crossover* (probCO), maka kedua induk dilakukan operasi pindah silang (*crossover*). tetapi jika bilangan p yang dibangkitkan lebih dari atau sama dengan probCO, maka tidak dilakukan operasi mutasi.

Teknik crossover yang digunakan adalah teknik *order crossover* (OX) yang diperkenalkan oleh Davis (Tanjung, 2010). Teknik OX diawali dengan membangkitkan dua bilangan acak. Kemudian gen yang berada diantara kedua bilangan acak akan disalin ke keturunan *(offspring)* dengan posisi yang sama. Langkah berikutnya untuk mendapatkan keturunan pertama adalah mengurutkan gen yang berada pada induk kedua dengan urutan gen yang berada pada posisi setelah bilangan acak kedua diikuti dengan gen yang berada pada posisi sebelum bilangan acak pertama dan diakhiri dengan gen yang berada pada posisi diantara kedua bilangan acak. Gen yang telah diurutkan tersebut dibandingkan dengan keturunan pertama. Apabila gen tersebut ada pada keturunan kedua maka abaikan gen tersebut dari urutan itu. Kemudian masukkan urutan yang baru saja didapat pada keturunan dengan cara memasukkan urutan gen pada posisi setelah bilangan acak kedua terlebih dahulu dan sisanya dimasukkan pada posisi sebelum bilangan acak pertama. Begitu juga untuk menghasikan keturunan kedua.

Contoh 2.4 *order cross over*

Dari 2 induk diketahui:

p_1 = (1 2 3 | **4 5 6 7** |8 9)

p_2 = (4 5 2 | **1 8 7 6** |9 3)

Dibangkitkan 2 bilangan acak sebelum gen induk-1 dan setelah gen induk-1. Hal yang sama juga dilakukan untuk induk-2. Didapatkan keturunan dengan gen yang sama:

o_1 = (x x x | **4 5 6 7** |x x)

o_2 = (x x x | **1 8 7 6** |x x)

Langkah berikutnya untuk mendapatkan keturunan pertama adalah mengurutkan gen yang berada pada induk kedua dengan urutan gen yang berada pada posisi setelah bilangan acak kedua diikuti dengan gen yang berada pada posisi sebelum bilangan acak pertama dan diakhiri dengan gen yang berada pada posisi diantara kedua bilangan acak.

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

Kemudian gen yang telah diurutkan tersebut dibandingkan dengan keturunan pertama. Apabila gen tersebut ada pada keturunan kedua maka abaikan gen tersebut dari urutan itu.

Kemudian masukkan urutan yang baru saja didapat pada keturunan dengan cara memasukkan urutan gen pada posisi setelah bilangan acak kedua terlebih dahulu dan sisanya dimasukkan pada posisi sebelum bilangan acak pertama. Begitu juga untuk menghasikan keturunan kedua. Keturunan 1 diperoleh:

o_1 = (x x x | 4 5 6 7 |x x)

o_1 = (2 1 8 | 4 5 6 7| 9 3)

dengan jalan yang sama buat o_2 sehingga :

o_2 = (x x x | 1 8 7 6 |x x)

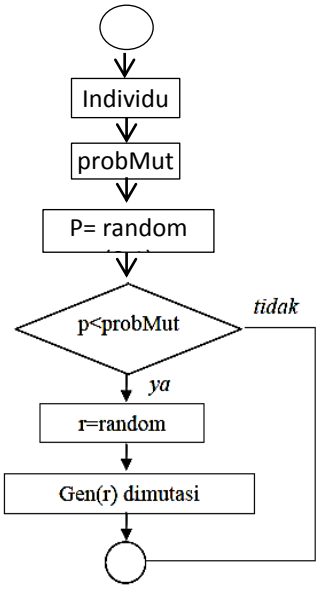
o_2 = (3 4 5 | 1 8 7 6 |9 2)

Keterangan:
p_1 = Induk 1
p_2 = Induk 2
o_1 = Keturunan 1 (anak ke-1)
o_2 = Keturunan 2 (anak ke-2)

f. Mutasi *(Mutation)*

Mutasi merupakan proses untuk mengubah nilai dari satu atau beberapa gen dalam suatu kromosom. Operasi mutasi yang dilakukan pada kromosom dengan tujuan untuk memperoleh kromosom-kromosom baru sebagai kandidat solusi pada generasi mendatang dengan *fitness* yang lebih baik, dan lama-kelamaan menuju solusi optimum yang diinginkan. Akan tetapi, untuk mencapai hal ini, penekanan *selektif* juga memegang peranan yang penting. Jika dalam proses pemilihan kromosom-kromosom cenderung terus pada kromosom yang memiliki *fitness* yang tinggi saja, konvergensi prematur akan sangat mudah terjadi (Murniati, 2009: 24).

Secara skematis proses mutasi dapat digambarkan sebagai berikut.



Gambar 2.5 Sistematika Proses Mutasi

Dari gambar 2.5 di atas, jika p merupakan bilangan random yang dibangkitkan kurang dari probabilitas mutasi (probMut) maka individu hasil *crossover* dilakukan proses mutasi Sedangkan jika bilangan p yang dibangkitkan lebih dari atau sama dengan probMut, maka individu hasil *crossover* tidak dilakukan proses mutasi.

Teknik *swapping mutation* diawali dengan memilih dua bilangan acak kemudian gen yang berada pada posisi bilangan acak pertama ditukar dengan gen yang berada pada bilangan acak kedua dalam probabilitas tertentu (Suyanto, 2005: 57).

Contoh 2.5 *swapping mutation*:

Individu = (1 **2** 3 4 5 6 **8** 9 7)

Memindahkan 8 ke 2, sehingga didapatkan individu baru:

Individu = (1 **8** 3 4 5 6 **2** 9 7)

g. Elitism

Elitism merupakan proses untuk menjaga agar individu bernilai fitness tertinggi tersebut tidak hilang selama evolusi (Kusumadewi, 2003: 112). Proses seleksi dilakukan secara random sehingga tidak ada jaminan bahwa suatu indvidu yang bernilai fitness tertinggi akan selalu terpilih. Walaupun individu bernilai *fitness* tertinggi terpilih, mungkin saja individu tersebut akan rusak (nilai *fitness*-nya menurun) karena proses pindah silang. Oleh karena itu, untuk menjaga agar individu bernilai *fitness* tertinggi tersebut tidak hilang selama evolusi, maka perlu dibuat satu atau lebih. Proses Elitism dilakukan dengan mendulpikat individu dengan

nilai fitness terbaik untuk dijadikan individu pertama pada generasi berikutnya

h. Pembentukan Populasi Baru

Proses membangkitkan populasi baru bertujuan untuk membentuk populasi baru yang berbeda dengan populasi awal. Pembentukan populasi baru ini didasarkan pada keturunan-keturunan baru hasil mutasi ditambah dengan individu terbaik setelah dipertahankan dengan proses *elitism*.

Setelah populasi baru terbentuk, kemudian mengulangi langkah-langkah evaluasi nilai fitness, proses seleksi dengan *truncation selection*, proses pindah silang, proses mutasi pada populasi baru untuk membentuk populasi baru selanjutnya.

5. Penelitian yang Relevan

Telah banyak penelitian tentang Algoritma genetik, antara lain " *Algoritma Genetik Dengan Metode Roullete Whell Selection dalam Pendistribusian Barang*" oleh Rudi Minaryo, dan "*Penerapan Algoritma Genetik pada Persoalan Pedagang Keliling (TSP)* " oleh Aulia Fitrah dkk. Pada dua penelitian tersebut algoritma genetik digunakan dalam menyelesaikan permasalahan *Travelling Salesman Problem* (TSP). Permasalahan yang dihadapi Aulia dkk adalah masalah TSP yang dimodelkan sedangkan pada penelitian Rudi permasalahan merupakan suatu masalah yang sesungguhnya.

Selain penelitian tentang algoritma genetik terdapat juga penelitian tentang Vehicle Routing Problem (VRP) . Salah satu contoh penelitian tentang VRP yaitu penelitian oleh Sri Nurhayanti yang berjudul

"*Perbandingan Metode Branch and Bound dengan Metode Clarke And Wright Savings untuk Menyelesaikan Masalah Distribusi Aqua Galon di PT. Tirta Investama*". Hasil penelitian ini didapatkan bahwa total jarak tempuh sebesar 147.7 Km dengan metode *Branch and Bound* dan 175.7 dengan metode *Clarke and Wright Savings*.

Penelitian tentang permasalahan VRP yang diselesaikan dengan algoritma genetik juga telah banyak dilakukan, salah satunya penelitian dengan judul " *Aplikasi Algoritma Genetik Hibrida pada Vechile Routing Problem With Time Windows*" oleh Sri Astuti. Pada penelitian yang dilakukan Sri Astuti dimulai dengan pembangkitan populasi awal yang dibagi menjadi 2 yaitu 50% dengan metode *Push Forward Insertion Heuristic* (PFIH) yang dilanjutkan dengan λ-*Intercharge*, dan 50% lainnya secara acak. Seleksi menggunakan seleksi rangking dan pindah silang dengan menggunakan *merge-heuristic crossover* serta *sequence based mutation* untuk mutasinya.

Pada penelitian ini memanfaatkan program Matlab Rudi Minaryo yang di modifikasi. Modifikasi yang dilakukan adalah merubah perhitungan nilai fitness karena pada penelitian yang dilakukan oleh Rudi Minaryo tidak terdapat pembagian rute, selain itu program pada penelitian ini metode seleksi yang digunakan berbeda yaitu *rank-based selection*. Program Matlab pada penelitian ini lebih sederhana dibandingkan dengan program Matlab yang digunakan Sri Astuti.

Ada persamaan dan perbedaan penelitian ini dengan penelitian sebelumnya. Persamaan pada skripsi Rudi Minaryo, Sri Astuti, dan

penelitian ini yaitu menggunakan algoritma genetik, sedangkan perbedaannya pada metode yang digunakan pada proses pencarian nilai fitness, seleksi, pindah silang, dan mutasi. Sedangkan pada penelitian Sri Nurhayati persamaan terdapat pada data yang digunakan dan perbedaan terdapat pada metode penyelesaian yang digunakan, pada penelitian ini menggunakan algoritma genetik sedangkan pada penelitian Sri Nurhayati menggunakan metode *Branch and Bound* dan *Metode Clarke And Wright Savings.* Karena data yang digunakan sama maka hasil penelitian ini dapat dibandingkan dengan hasil penelitian Sri Nurhayati.