PENERAPAN ALGORITMA GENETIKA DALAM MENEMUKAN RUTE TERPENDEK

Diana.Fallo[1)], Alb.Joko Santoso[2)], Djoko Budiyanto[3)]

[1]Magister Teknik Informatika, Fakultas Teknologi Industri, Universitas Atma Jaya Yogyakarta
Email :[1)] dianayani25@gmail.com, [2)]albjoko@mail.uajy.ac.id, [3)]Djoko@mail.uajy.ac.id

ABSTRACT

Shortest route searching is a problem often encountered in the transportation field. The function of genetic algorithm is to determine the optimal solution value to a problem with numerous solutions. In this research, there were 63 tests using 6 location spots. The test results using values population (10,15),crossover (0.2, 0.3, 0.5, 0.6, 0.7, 0.8), mutation (0.1, 0.05, 0.4, 0.7) dan iterations (200,300,400,500) showed that the shortest route was 239,45 km and the times was 6.65. Algorithm accuracy level of 85,7% and the route can be skipped is A-C-D-B-E-F-A.

Keywords:

Shortest path, genetic algorithm, *travelling salesman problem*

1 . Pendahuluan

Algoritma genetika berasal dari bahasa Yunani, 'genesis' yang berarti 'tumbuh' atau 'menjadi'. tujuannya adalah untuk menemukan soulsi perkiraan untuk masalah optimasi yang terlihat sederhana yang bekerja pada n kota yang berbeda untuk di jadikan tour. Dengan kata lain masalah ini berkaitan dengan pencarian rute yang meliputi seluruh kota sehingga jarak perjalanan dapat di minimalkan [6].

Algoritma genetika dikembangkan pada tahun 1960 oleh John Holland dengan mengambil prinsip Darwin yaitu mengubah populasi dari individu menjadi nilai *fitness* untuk menjadi generasi baru kemudian dilakukan *crossover* dan mutasi dan algoritma genetic berhasil menjawab berbagai masalah optimasi meskipun masih ada beberapa factor yang membatasi keberhasilan algoritma genetika [16 & 11]

Algoritma genetika memiliki fungsi yaitu mendapatkan nilai solusi optimal terhadap permasalahan yang mempunyai banyak kemungkinan solusi [10]. Hasil penentuan jarak terpendek akan menjadi pertimbangan dalam pengambilan keputusan untuk menunjukkan jalur yang akan ditempuh [12].

Penyelesaian *Travelling Salesman Problem* (TSP) adalah mencari lintasan terpendek kunjungan ke semua kota yang banyak di pakai dalam agoritma genetika. dengan lintasan terpendek tersebut maka waktu tempuh diharapkan juga kan menjadi lebih cepat. Syarat kerja TSP adalah -kota tersebut hanya dikunjungi sekali sebelum akhirnya kembali ke kota asal [1].

Tujuan utama dari penyelesaian problem ini adalah untuk mencari rute terpendek yang bisa diaplikasikan dalam jarak tempuh dan waktu tempuh. Dengan memanfaatkan teknolgi system informasi geografis maka diharapkan dapat memberikan informasi yang bermanfaat.

2. Metode Analisis Data

Algoritma genetika merupakan bagian dari evolusi yang terbagi dalam beberapa tahap [1,3, 13&14]

1. Inisialisasi /*encoding*
2. Evaluasi *Fitness*
3. Seleksi
4. *Crossover*
5. Mutasi

Langkah pertama dalam menggunakan algoritma genetika adalah menentukan gen. Defisini gen yaitu node-node yang akan menjadi tujuannya. Kumpulan gen akan membuat suatu populasi sehingga menjadi suatu solusi awal jalur penyelesaian. Skema *encoding* yang digunakan adalah *permutation encoding*. pada *permutation encoding* setiap kromosom terdiri atas deretan angka yang menyatakan posisi dalam suatu urutan. Nilai dalam suatu lokasi yang ada terdiri atas deretan angka yang menyatakan posisi dalam suatu urutan[2]. Sebagai contoh hasil *encoding,* berikut ini [5]

Kromosom[1] = [F B E C D]
Kromosom[2] = [F B D E C]
Kromosom[3] = [E B C F D]
Kromosom[4] = [C E B D F]
Kromosom[5] = [C D E F B]
Kromosom[6] = [B C F E D]

Tahap kedua adalah proses evaluasi. Tahap ini dilakukan untuk mencari nilai *fitness*. Individu yang bernilai *fitness* terbaik akan bertahan hidup sedangkan yang bernilai rendah akan mati [2].

Umumnya kromosom yang memiliki nilai *fitness* tinggi akan bertahan dan berlanjut ke generasi berikutnya. Sebagai contoh perhatikan di bawah ini [5].

Jarak[1]
=AF+FB+BE+EC+CD+DA=108+88+73+61+ 25+49=404
Jarak[2]
=AF+FB+BD+DE+EC+CA=108+88+38+43+61+33=371
Jarak[3]
=AE+EB+BC+CF+FD+DA=89+73+13+76+62+49=359
Jarak[4]
=AC+CE+EB+BD+DF+FA=33+61+73+38+62+108=375
Jarak[5]
=AC+CD+DE+EF+FB+BA=33+25+43+22+88+29=240
Jarak[6]
= AB+BC+CF+FE+ED+DA=29+13+76+22+43+49=232

Selanjutnya hitung yang jarak ada untuk mendapatkan nilai setiap fitness mengunakan rumus (1/Jarak)[5 &10]. Contohnya sebagai berikut

Fitness[1]= 1/Jarak[1])= 1/404=0.00247
Fitness[1]= 1/Jarak[2])= 1/371=0.00269
Fitness[1]= 1/Jarak[3])= 1/359=0.00278
Fitness[1]= 1/Jarak[4])= 1/375=0.00266
Fitness[1]= 1/Jarak[5])= 1/240=0.00416
Fitness[1]= 1/Jarak[6])= 1/232=0.00431
Total Fitness = 0.00247 + 0.00269+ 0.00278+ 0.00266 +0.00416 + 0.00431 =0.01907

Langkah ketiga adalah proses seleksi. Proses seleksi dalam penelitian ini yaitu menggunakan *Tournament Selection. S*eleksi Turnamen bekerja dengan cara memilih beberapa jumlah t individu secara acak dari populasi dan mengopy individu terbaik dari grup ini ke dalam pupolasi,dan ulangi sebanyak N kali. Seringkali dalam turnamen yang dipegang hanya antara dua individu turnamen biner saja, tetapi umumnya memungkinkan untuk memegang sembarang ukuran kelompok t yang disebut dengan ukuran turnamen[15&17].

Metode *tournament Selection* ini mula-mula ditetapkan suatu nilai tour *t* untuk individu-individu yang dipilih secara random dari suatu populasi. Individu-individu yang terbaik dalam kelompok ini akan dipilih sebagai induk. Parameter yang digunakan pada metode ini adalah ukuran tour yang bernilai antara 2 sampai N (jumlah individudalam suatu populasi) [17].

```
Binary_tournament_selection (p,fv,n,N
S= new population of size N;
For i=1 to n
A=rand (1,n);
B=rand (1,n-1);
If (b>=a) b++;
If (fv[a]>fv[b])
S[i]=P[a];
Else
S[i]=P[b];
Return S;
```

Gambar 1. *Pseducode tournament selection* [17]

Langkah ke empat adalah proses *crossover*. *Order crossover* ini diperkenalkan oleh Davis. Teknik ini diawali dengan membangkitkan dua bilangan acak. Kemudian gen yang berada pada *parent* kedua bilagan acakan akan disalin ke *offspringi* dengan posisi sama. Berikutnya untuk mendapatkan *offspring* pertama adalah mengurutkan gen yang berada pada *parent* kedua dengan urutan gen yang berada pada posisi stetelah bilangan acak kedua diikuti dengan gen yang berada pada posisi sebelum bilangan acak pertama dan diakhiri dengan gen yang berada pada posisi diantara ke dua bilangan acak[10].

Gen yang telah diurutkan tersebut dibandingkan dengan *offspring* pertama. Apabila gen tersebut ada pada *offspring* ke dua maka abaikan gen tersebut dari urutan itu. Kemudian masukan urutan yang baru saja di dapat pada *offspring* dengan cara memasukkan pada posisi sebelum bilangan acak pertama [4].

Sebuah contoh diketahui *parent* adalah 678 ditambahkan ke dalam anak/keturunan. Untuk *parent* 9 yang tidak ikut pindah menduduki posisi pertama posisi berikutnya diikuti oleh 8 namun karena 8 sudah di pindahkan maka ini dilewati dan diisi oleh 5 dan seterusnya. Keturunan yang lain sampai semua terisi tanpa ada posisi yang hilang atau digandakan.

Perhatikan contoh perhitungan *crossover* pada gambar 6.

Parents

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

9	8	7	6	5	4	3	2	1

Offspring

					6	7	8	

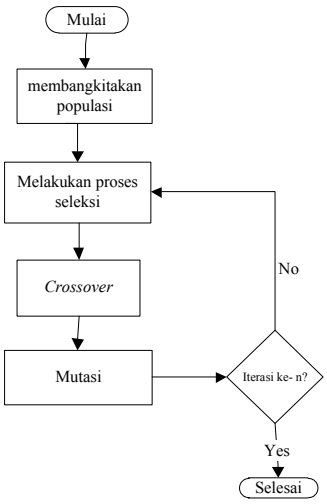
9	**5**	**4**	**3**	**2**	6	7	8	1

Gambar 2. Contoh perhitungan *crossover* [9]

Langkah ke lima yaitu proses mutasi. Teknik pertukaran yang digunakan dalam penelitian ini adalah *swapping mutation*. Menurut Hardianti & Purwanto (2013) dalam penelitiannya yang berjudul *Penerapan Algoritma Genetika dalam Penyelesain Travelling Salesman Problem With Predence Constraints (TSPPC)* [8] mengemukakan bahwa cara kerja metode ini adalah sebagai berikut :

a. Membangkitkan bilangan acak sebanyak total bit (jumlah kromosom dikalikan dengan *pop_size*) untuk menetukan gen yang termutasi. Bilangan acak yang dibangkitan adalah real antara 0 sampai 1.
b. Mencari letak bilangan acak yang kurang *probabilitas* mutasi.
c. Menukarkan gen dengan bilangan acak kurang dari *probabilitas* mutasi dengan gen sesudahnya.

Berikut ini diagram *flowchart* algoritma genetika, telihat pada gambar 3.

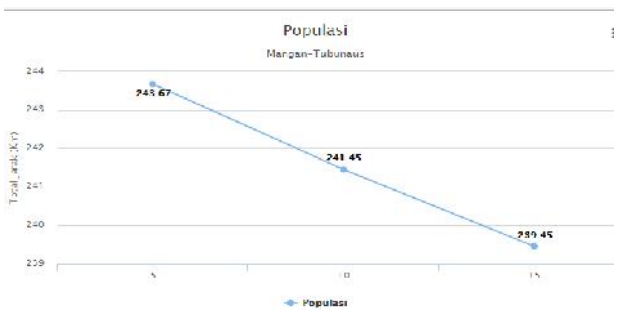


Gambar 3. Diagram Flowchart Algoritma Genetika [7]

Proses dimulai dari inisialisasi populasi, evaluasi *chromosome*, seleksi, *crossover* dan mutasi. Jika generasi baru telah di temukan dan *output*nya cukup dekat atau sama dengan jawaban yang diinginkan maka masalah selesai. Namun jika tidak menemukan mana generasi baru akan melalui proses ulang lagi sampai solusi tercapai. Pemberian nilai pada setiap parameter sangat berpengaruh terhadap jarak yang dihasilkan. [14&16].

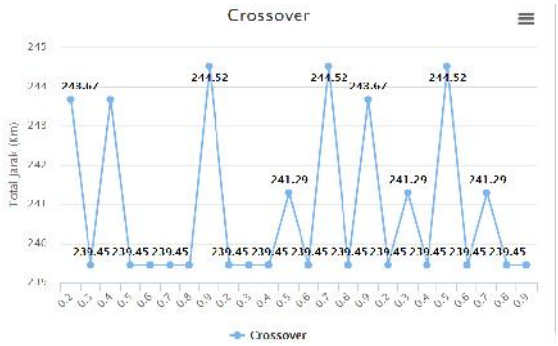
3. Pembahasan

Uji coba dalam penelitian ini menggunakan program berbasis *open source* yaitu Javascript, JQuery. Pengujian dilakukan dengan menggunakan beberapa parameter nilai yang berbeda yaitu populasi {5,10,15}, mutasi {0.01, 0.05, 0.1,0.05, 0.4, 0.7} *crossover* {0.1-0.9} dan *max generations* {20-500}



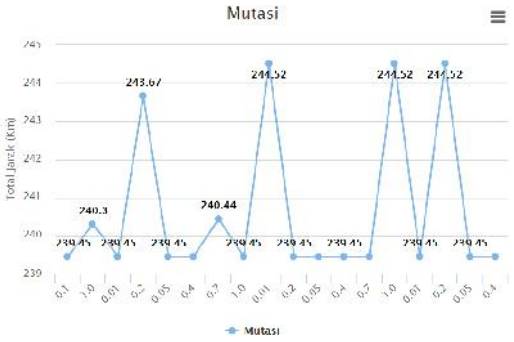
Gambar 4. Grafik nilai populasi

Nilai populasi pada pengujian ini adalah {5,10,15} nilai mutasi {0.1} nilai *crossover* {0.1} *iterasi* {50} pengujian di lakukan sebanyak tiga kali dan di temukan bahwa semakin besar nilai populasi maka semakin baik jarak yang diperoleh.



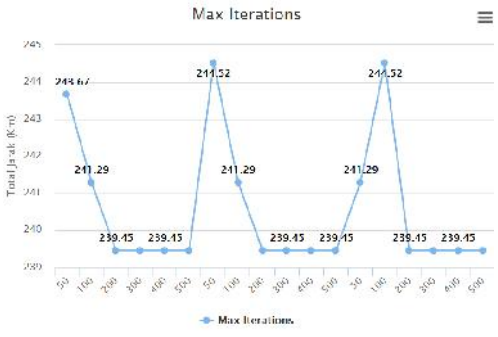
Gambar 5. Grafik nilai *crossover (pc)*

Uji coba pada nilai *crossover* adalah {0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9} nilai mutasi {0.1} nilai iterasi {50} dan populasi {5,10,15}. Dari hasil uji coba di temukan jarak yang muncul sama sebanyak 12 kali.



Gambar 6. Grafik nilai mutasi (pm)

Uji coba selanjutnya menggunakan nilai mutasi yaitu {0.01, 0.02, 0.05,0.1, 0.2, 0.4, 0.7} nilai *crossover* {0.1} *iterasi*{50}. Di temukan jarak yang sama muncul sebanyak 15 kali.



Gambar 7. Grafik nilai iterasi

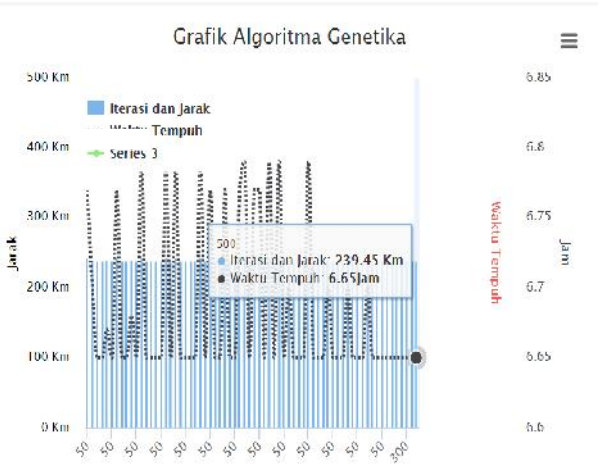
Pengujian nilai iterasi adalah {50,100,200,300,400,50} nilai mutasi {0.1} nilai *crossover* {0.1} dan populasi {5,10,15} muncul jarak yang sama sebanyak 12 kali.

Selanjutnya pengujian dilakukan secara keseluruhan dan terlihat pada tabel 1 dimana hanya di tampilkan 20 kali pengujian dari total 63 kali pengujian.

Tabel 3.1 Pengujian Algoritma Genetika

Populasi	pc	*pm*	*Iterasi*	Jarak (Km)	Waktu (Jam)	Rute
5	0.1	0.1	50	243,67	6,77	A-B-C-D-F-E-A
10	0.1	0.1	50	241,45	6,70	A-B-C-E-F-D-A
15	0.1	0.1	50	239,45	6,65	A-C-D-B-E-F-A
5	0.1	1.0	50	239,45	6,65	A-B-C-F-E-D-A
5	0.1	0.01	50	240,30	6,67	A-C-D-B-E-F-A
5	0.1	0.2	50	239,45	6,65	A-C-D-B-E-F-A
5	0.1	0.05	50	243,67	6,77	A-B-C-D-F-E-A
5	0.1	0.4	50	239,45	6,65	A-C-D-B-E-F-A
5	0.1	0.7	50	239,45	6,65	A-C-D-E-F-B-A
5	0.1	1.0	50	239,45	6,65	A-C-D-B-E-F-A
10	0,1	0,2	50	239,45	6,65	A-C-D-B-E-F-A
10	0,1	0,3	50	239,45	6,65	A-C-D-B-E-F-A
10	0,1	0,4	50	243,67	6,77	A-D-E-F-C-B-A
10	0,1	0,5	50	244,52	6,79	A-B-D-E-F-C-A
10	0,1	0,6	50	239,45	6,65	A-C-D-B-E-F-A
10	0,1	0,7	50	243,67	6,77	A-B-C-D-F-E-A
10	0,1	0,8	50	243,67	6,77	A-B-C-D-F-E-A
10	0,1	0,9	50	239,45	6,65	A-C-D-B-E-F-A
15	0,1	0,2	100	239,45	6,65	A-C-D-B-E-F-A
15	0,1	0,2	200	239,45	6,65	A-C-D-B-E-F-A

Berdasarkan uji coba pada tabel 1 maka berikut ini merupakan grafik pengujian secara keseluruhan algoritma genetika pada enam titik lokasi. Terlihat pada gambar 8 dibawah ini



Gambar 8. Grafik pengujian algoritma genetika

Pengujian dilakukan sebanyak 63 kali dan muncul jarak yang sama sebanyak 54 kali yaitu 239.45Km dengan waktu tempuh 6.65 Jam dengan tingkat akurasi mencapai 85,7%.

4. Kesimpulan

Terdapat beberapa kesimpulan dalam penelitian ini yaitu:

1. Jarak terpendek yang paling banyak muncul dalam pengujian ini adalah 239.45 Km denganwaktu tempuh 6.65 Jam dan rute terpendeknya
2. Tingkat akurasi algoritma mencapai 85,7% sehingga terbukti bahwa algoritma ini dapat bekerja dengan baik dalam menemukan rute terpendek.
3. Parameter terbaik dalam uji coba yaitu nilai populasi (10,15), *crossover* (0.2, 0.3, 0.5, 0.6, 0.7, 0.8), mutasi (0.1, 0.05, .0.4, 0.7) dan iterasi dari 200, 300, 400, 500 mampu menghasilkan jarak dengan lebih cepat.
4. Semakin besar nilai populasi dan iterasi yang diberikan kemungkinan untuk memperoleh rute terpendek semakin cepat dan lebih optimal.
5. Kekurangan dari algoritma ini adalah penggunaan memory yang boros dan algoritma ini cenderung berubah-ubah/ lebih bervariasi terhadap nilai yang berikan.
6. Jarak yang hasilkan adalah yang optimal namun secara kenyataannya belum tentu optimal algoritma ini bisa menghasilkan jarak yang lebih panjang dari jarak sebenarnya.

REFERENSI

[1] Asim, M., Gopalia, & Shivalika, S., 2014. "Travelling Salesman Problem Using Genetic Algorithm",International Journal of Lastest Trends in Engineering and Technology (IJLTET), 3(3), p.183.

[2] Baharudin, A., Mazharuddin , A. & Pratomo, B.A., 2012. Travelling Salesman Problem Menggunakan Algoritma Genetika Via GPS Berbasis Android. Surabaya: Institut Teknologi Sepuluh November.

[3] Dwivedi, V., Chauhan, T., Saxena, S. & Agrawal, P., 2012. "Travelling Salesman Problem using Genetic Algorithm" International Journal of Computer Applications (IJCA).

[4] Fachrurrazi, S., 2103. *Penerapan Algoritma Genetika Dalam Optimasi Pendistribusian Pupuk di PT Pupuk Iskandar Muda Aceh Utara*. Aceh Utara: Program Studi Teknik Informatika Universitas Malikussaleh Reuleut.

[5] Fitrah, A., Zaky , A. & Fitrasani, 2006. "Penerapan Algoritma Genetika pada Persolana Pedagang Keliling (TSP)". Institur Teknologi Bandung.

[6] Gupta , A. & Khurana, , 2012. "Study of Travelling Salesman Problem Using Genetic Algorithm", International Journal of Management, IT and Engineering , 2(5).

[7] Gupta, S. & Panwar, P., 2013. "Solving Travelling Salesman Problem Using Genetic Algorithm", International Journal of Advanced Research in Computer Science and Software Engineering, 3(6).

[8] Hardianti, Y. & Purwanto, 2013. Penerapan Algoritma Genetika dalam Travelling Salesman Problem With Precedence Constraints (TSPPC). *Jurnal Online*.

[9] Jacobson, L., 2012. *Applying a genetic algorithm to the traveling salesman problem*. [Online] Available at: http://www.theprojectspot.com/tutorial-post/applying-a-genetic-algorithm-to-the-travelling-salesman-problem/5 [Accessed Monday Mei 2015].

[10] Joni, I.M.A.B. & Nurcahyawati, V., 2012. "Penentuan Jarak Terpendek Pada Jalur Distribusi Barang di Pulau Jawa dengan Menggunakan Algoritma Genetika", Jurnal Nasiona Pendidikan Teknik Informatika (JANAPATI) , 1.

[11] Khan, H.F., Khan, N., Inayatulla, S. & Nizami, A.S.T., 2009. Solving ISP Problem by Using Genetic Algorithm. *International Journal of Basic & Applied Sciences IJBAS-IJENS*, 09.

[12] Lestari, U., Widyastuti, N. & Arghina, D., 2014. Impelementasi Algoritma Genetika pada Penjadwalan Perkuliahan. *Prosiding Seminar Nasional Aplikasi Sains dan Teknologi (SNAST)* .

[13] Mutakhiroh , I., Saptono, , Hasanah , N. & Wiryadinata, R., 2007. "Pemanfaatan metode Heuristik dalam Pencarian Jalur Terpendek dengan Algoritma Semut dan Algoritma Genetika", Seminar Nasional Aplikasi Teknologi Informasi (SNATI).

[14] Philip, A., Taofiki, A.A. & Kehinde, O., 2011. "Genetic Algorithm for Solving Travelling Salesman Problem", International Journal od Advanced Computer Science and Applications.

[15] Razali, N.M. & Geraghty, J., 2011. "Genetic Algorithm Performance with Different Selection Strategis in Solving TSP". Proceedings of the Worls Congress in Engineeting .

[16] Rexhepi, A., Maxhuni, A. & Dika, A., 2013. Analysis of the impact of parameters values on the Genetic Algorithm for TSP. *International Journal of Computer Science Issues*, 10.

[17] Siami, I.I., 2012. *Metode Seleksi Algoritma Genetika*. [Online] Available at: https://www.scribd.com/doc/97492749/Metode-Selection [Accessed Monday April 2015].

Nama Penulis

Diana Fallo, saat ini sedang menempuh pendidikan Magister Teknik Informatika pada Universitas Atma Jaya Yogyakarta.

Alb, Joko Santoso. Bekerja sebagai Pengajar pada Universitas Atma Jaya Yogyakarta. Sekaligus menjadi Pembimbing I bagi Penulis dalam penulisan tesis.

Djoko Budiyanto, Bekerja sebagai Pengajar pada Universitas Atma Jaya Yogyakarta. Sekaligus menjadi Pembimbing II bagi Penulis dalam penulisan tesis.