ANALISIS ALGORITMA GENETIKA TANPA OPERASI *CROSSOVER* PADA KASUS *TRAVELLING SALESMAN PROBLEM* DENGAN METODE 2 TAHAP MUTASI

Sean Coonery Sumarta
Program Studi Informatika, Fakultas Teknologi Informasi, Universitas Atma Jaya Makassar
Alamat email: sean_chiko@yahoo.com

ABSTRACT

The TSP problem can resolved using Genetic Algorithms. In other previous research, crossover is one process that can be eliminated from the genetic algorithm to get better optimization. This research tried to solve TSP by using genetic algorithms without using the crossover. To improve optimization by applying the method of mutation with 2 phases. Mutation operator used the inversion mutation, exchange mutation, and insertion mutation . The results using these methods is the average error percentage of optimum distance <10%.

Keywords: *Travelling Salesmen Problem, Genetic Algorithm, Two Step Mutattion*

1. PENDAHULUAN

Salah satu permasalahan dalam bidang transportasi darat adalah mencari suatu rute perjalanan terpendek yang dapat di tempuh dari titik awal keberangkatan menuju titik akhir (tujuan), dengan harapan biaya perjalanan yang dikeluarkan dan waktu tempuh perjalanan seminimum mungkin. Masalah seperti ini dikategorikan dalam suatu permasalahan kombinatorial yang lebih dikenal dengan *Traveling Salesman Problem* (TSP). Terdapat beberapa pendekatan yang dapat digunakan dalam mencari solusi optimum dalam menjawab masalah TSP, diantaranya Algoritme Genetika, Pendekatan secara Stokastik, Neural Network dan pemograman Linier [1].

Algoritme Genetika adalah algoritme optimasi modern berdasarkan evolusi biologis, yang meminjam hukum evolusi biologis yaitu untuk mencapai kelangsungan hidup yang paling cocok melalui pembiakan, warisan, diferensiasi dan persaingan, untuk mendekati solusi optimal secara bertahap. Algoritme Genetika adalah salah satu algoritme matang untuk memecahkan masalah *Nondeterministic Polynomial* (NP) saat ini, yang memiliki aplikasi luas dalam teori optimasi yang modern[2]. Algoritme ini memanfaatkan proses seleksi alamiah yang dikenal dengan proses evolusi. Dalam proses evolusi, individu secara terus-menerus mengalami perubahan gen untuk menyesuaikan dengan lingkungan hidupnya. "Hanya individu-individu yang kuat yang mampu bertahan". Proses seleksi alamiah ini melibatkan perubahan gen yang terjadi pada individu melalui proses perkembangbiakan. Proses perkembangbiakan ini didasarkan pada analogi struktur genetik dan perilaku kromosom dalam populasi individu[3].

Algoritme genetika menggunakan teknik dan proses yang terinspirasi dari evolusi biologi untuk memecahkan masalah optimasi yang kompleks, terdapat beberapa seleksi diantaranya *natural selection, crossover and mutation* diterapkan untuk mendapatkan nilai baru dan menemukan solusi yang optimal. Algoritme genetika biasa digunakan untuk menyelesaikan berbagai permasalahan kompleks pada bidang fisika, biologi, ekonomi, sosiologi.

Beberapa penelitian yang dilakukan sebelumnya mengenai operator *crossover* dan mutasi cukup banyak diantaranya adalah referensi [4] yang melakukan evaluasi terhadap pasangan *crossover* dan mutasi yang memiliki nilai fitness yang terbaik serta penelitian yang dilakukan dalam referensi [5] yang menetapkan operator terbaik dalam tiap-tiap proses pada algoritme genetika dengan memasangkan tiap-tiap metode pada masing-masing operator algoritme genetika dan mengevaluasinya.

Modifikasi Algoritme genetika telah banyak dilakukan oleh para peneliti dengan tujuan yang berbeda dan disesuikan dengan permasalahan yang ditemui. Modifikasi

terhadap Algoritme genetika dengan melihat kelemahan pada proses *crossover* seperti yang dilakukan oleh [6] dengan melakukan pemrosesan paralel pada proses *crossover* untuk memperoleh waktu koputasi yang lebih cepat. Modifikasi yang dilakukan oleh [7] dengan menambahkan proses tertentu pada proses *crossover* untuk mendapatkan nilai optimum yang lebih baik. Dan yang dilakukan oleh [8] dan [9] dengan cara mengeliminasi proses *crossover* dari Algoritme Genetika dan mengandalkan proses mutasi untuk meliharkan individu baru dengan nilai optimasi yang lebih baik.

Proses mutasi Algoritme Genetika pada penerapannya dalam kasus TSP ada beberapa metode antara lain : *Inversion Mutation*, *Exchange Mutation* dan *Insertion Mutation*.

Penelitian ini fokus pada operator mutasi untuk menyelesaikan masalah TSP dengan menggunakan kombinasi dari 3 operator mutasi yaitu *Inversion Mutation*, *Exchange Mutation* dan *Insertion Mutation*.

2. TINJAUAN PUSTAKA

2.1 *Traveling Salesman Problem*

Permasalahan matematika tentang TSP dikemukakan pada tahun1800 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.L. Loyd dan R. J. Wilson di Oxford, tahun 1976. [10]. Bentuk umum dari TSP pertama dipelajari oleh para matematikawan mulai tahun 1930. Diawali ole Karl Menger di Vienna dan Harvard. Setelah itu permasalahan TSP dipublikasikan oleh Hassler dan Merrill Flood di Princeton. Penelitian secara detail dari hubungan Menger dan Whitney [11].

TSP merupakan suatu permasalahan optimasi dengan tujuan menemukan jalur perjalanan dengan biaya evaluasi minimum. Biaya evaluasi yang dimaksud dapat berupa jarak, waktu, bahan bakar, kenyamanan dan sebagainya. Dalam TSP, *salesmen* harus mengunjungi setiap kota yang ada tepat satu kali dan kembali ke kota asal.

Pada kasus TSP terdapat 2 jenis kasus TSP yaitu TSP Asimetris dan TSP Simetris. Perbedaan 2 jenis kasus TSP dapat dijelaskan sebagai berikut:

a. TSP Asimetris

Pada TSP Asimetris, biaya dari *node* 1 ke *node* 2 tidak sama dengan biaya *node* 2 ke *node* 1 yang dinyatakan dalam Persamaan 2.1 sebagai berikut :

$$n_{12} \neq n_{21} \quad (1)$$

Jumlah jalur yang mungkin adalah permutasi dari jumlah *node* dibagi dengan jumlah *node*. Hal ini dapat dipahami secara siklus, sebuah jalur dengan urutan $n_{123} = n_{231} = n_{312}$, tetapi jalur $n_{123} \neq n_{321}$. Jadi apabila terdapat 10 *node*, maka jalur yang mungkin untuk TSP Asimetris dapat dihitung dengan menggunakan Persamaan 2.2:

$$P_{10}^{10} = \frac{10!}{10} = 362.880\ jalur \quad (2)$$

b. TSP Simetris

Pada TSP Simetris, biaya dari *node* 1 ke *node* 2 sama dengan biaya dari *node* 2 ke *node* 1 yang dinyatakan dalam Persamaan 2.3.

$$n_{12} = n_{21} \quad (3)$$

Jumlah jalur yang mungkin adalah permutasi dari jumlah *node* dibagi dengan 2 kali jumlah *node*. Hal ini dapat dipahami secara siklus, sebuah jalur dengan urutan $n_{123} = n_{231} = n_{312} = n_{321}$. Jadi apabila terdapat 10 *node* , maka jalur yang mungkin untuk TSP Simetris dapat dihitung dengan menggunakan Persamaan 2.4 sebagai berikut :

$$p_{10}^{10} = \frac{10!}{2 \times 10} = 181.440\ jalur \quad (4)$$

Pada TSP hasil yang ingin didapatkan adalah menemukan jalur terpendek yang saling terhubung dari *n* kota. Tiap kota hanya boleh dikunjungi sekali. Jarak antar kota *i* dan *j* dihitung dengan Persamaan *Euclidean*. Persamaan *Euclidean* dalam bentuk matematika dapat dilihat pada Persamaan 2.5 sebagai berikut :

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \quad (5)$$

Dengan :

d_{ij} = jarak kota *i* dengan kota *j*
x_i, y_i = koordinat (x,y) kota *i*
x_j, y_j = koordinat (x,y) kota *j*

TSP dapat direpresentasikan ke dalam permasalahan graf. Tiap kota diwakili oleh *node* dalam koordinat *Cartesian*. Tiap *node* mempunyai koordinat (x,y) dan semua *node* tersebut saling terhubung satu dengan lainnya dengan jarak masing-masing *node* dapat dihitung dari koordinat-koordinat *node* masing-masing. Jika ada *node* yang tidak terhubung, maka *node* tersebut dihubungkan dan diberi nilai jarak yang nantinya bisa digunakan sebagai solusi optimal.

2.2 Algoritme Genetika

Algoritme Genetika (*Genetic Algorithms*) adalah algoritme yang memanfaatkan proses seleksi alamiah yang dikenal dengan proses evolusi. Pada dasarnya, evolusi dilakukan karena individu harus menyesuaikan diri dengan perubahan lingkungan yang terjadi seiring perkembangan waktu. Penyesuain diri dilakukan dengan melakukan perubahan gen melalui proses perkembang-biakan

Terdapat empat kondisi yang sangat mempengaruhi proses evaluasi [12], adalah :

a. Kemampuan individu untuk melakukan reproduksi.
b. Keberadaan populasi individu yang melakukan reproduksi.
c. Keberagaman individu dalam suatu populasi.
d. Perbedaan kemampuan untuk bertahan.

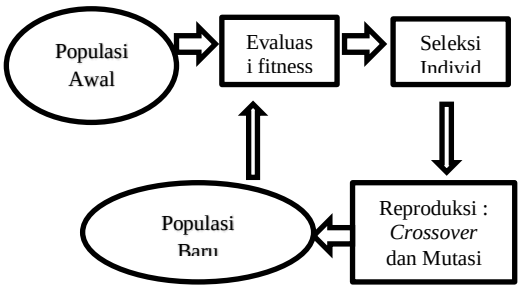
Individu yang lebih kuat akan memiliki kemampuan bertahan dan reproduksi yang lebih tinggi dibandingkan dengan individu yang lebih lemah. Pada kurun waktu tertentu, populasi akan lebih banyak memuat individu yang lebih kuat.

Algoritme genetika pertama kali dikenalkan oleh Jhon Holland dari Universitas Michigan pada tahun 1975[13]. Jhon Holland meyatakan bahwa setiap masalah yang berbentuk adaptasi atau penyesuaian diri, alami maupun buatan, dapat diformasikan dala terminologi genetika. Sejak pertama kali dikenalkan, Algoritme genetika telah dipelajari, diteliti dan diaplikasikan secara luas pada berbagai bidang. Algoritme Genetika banyak digunakan pada masalah praktis yang berfokus pada pencarian parameter-parameter optimal. Hal ini membuat banyak orang mengira bahwa Algoritme Genetika hanya digunakan untuk masalah optimasi. Namun pada kenyataanya, Algoritme Genetika juga memiliki kinerja yang bagus untuk menyelesaikan masalah-masalah selain optimasi.

Algoritme genetika yang dibuat Holland merupakan sebuah metode untuk memisahkan satu populasi kromosom (terdiri dari bit-bit 1 dan 0) ke baru dengan menggunakan ”seleksi alam” dan operator genetik seperti *crossover* dan *mutation*. *Crossover* menukar bagian kecil dari dua kromosom, mutation mengganti secara acak nilai gen dibeberapa lokasi pada kromosom. Dasar teori inilah yang menjadi dasar kebanyakan program yang menggunakan algoritme genetika pad saat ini.

Algoritme genetika digunakan untuk mendapatkan solusi yang tepat untuk masalah optimasi dari satu variabel atau multi variabel. Berbeda dengan teknik pencarian konvensional, algoritme genetika bermula dari himpunan solusi yang dihasilkan acak, himpunan ini disebut populasi. Sedangkan setiap inidividu dalam populasi disebut kromosom yang merupakan representasi dari solusi. Kromosom-kromosom berevolusi dalam suatu proses iterasi yang berkelanjutan yang disebut generasi. Pada setiap generasi, kromosom dievaluasi berdasarkan fungsi evaluasi. Setelah beberapa generasi maka algoritma genetika akan konvergen pada kromosom terbaik, yang diharapkan merupakan solusi optimal[13].

Siklus dari algoritme genetika menurut [13], dapat dilihat pada gambar 1 sebagai berikut :



Gambar 1. Siklus algoritme Genetika.

2.3 *Inversion Mutation*

Inversion mutation dilakukan dengan membangkitkan 2 buah nilai random

[0,gen], kemudian membalikkan posisi kota yang berada dalam rentang 2 nilai random tersebut.
Contoh:
Individu dari sebuah urutan kota (P)
P = (5 7 1 4 2 8 3 6 9 0)
gen = 10.
Hasil dari random nilai adalah 2 dan 6.
P = (5 |**7 1 4 8 3**| 6 9 0)
C = (5 **3 8 4 1 7** 6 9 0)
C adalah individu hasil dari mutasi.

2.4 *Exchange Mutation*

Inversion mutation dilakukan dengan membangkitkan 2 buah nilai random [0,gen], kemudian tukar 2 posisi kota yang sesuai dengan 2 random tersebut.
Contoh:
Individu dari sebuah urutan kota (P)
P = (5 7 1 4 2 8 3 6 9 0)
Hasil dari random nilai adalah 2 dan 6.
P = (5 **7** 1 4 8 **3** 6 9 0)
C = (5 **3** 1 4 8 **7** 6 9 0)

2.5 *Insertion Mutattion*

Insert mutation dilakukan dengan membangkitkan 2 buah nilai random [0,gen], kemudian tukar posisi kota titik nilai random pertama ke posisi kota nilai random kedua, kota sesudah kota posisi nilai random pertama akan mengisi posisi nilai random pertama dan seterusnya sampai pada posisi nilai random kedua.
Contoh:
Individu dari sebuah urutan kota (P)
P = (5 7 1 4 2 8 3 6 9 0)
Hasil dari random nilai adalah 2 dan 6.
P = (5 **7** 1 4 8 3 6 9 0)
C = (5 1 4 8 3 **7** 6 9 0)

3. METODOLOGI PENELITIAN

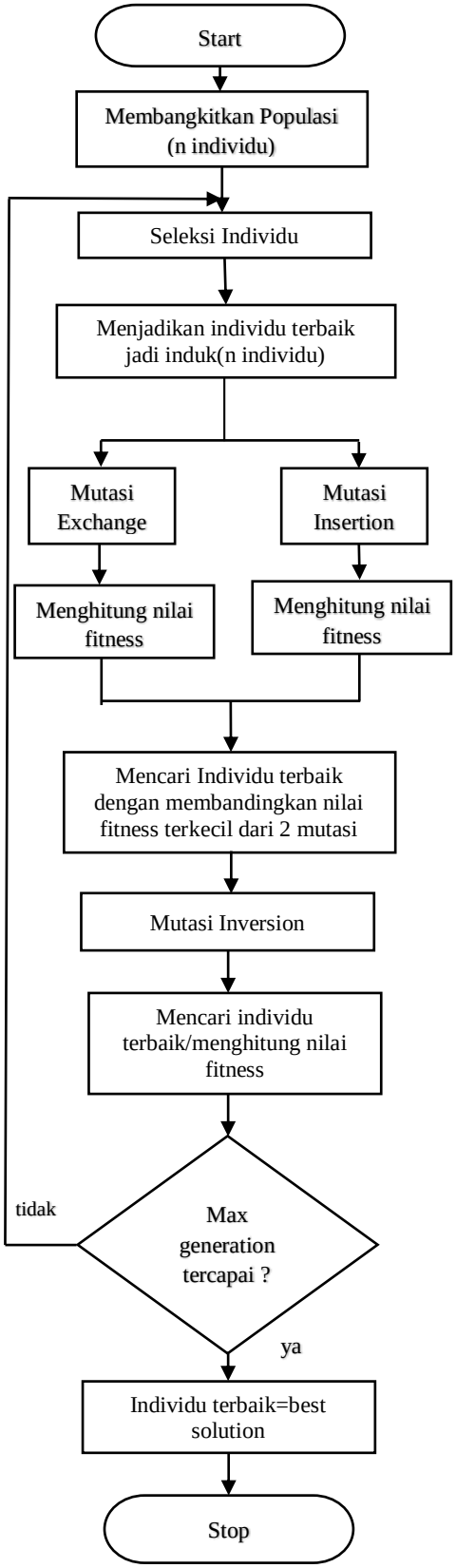
Pada penelitian ini, algoritme genetika yang digunakan seperti yang telah dikemukakan oleh [13] namun mengeliminasi proses *crossove* seperti yang dapat dilihat pada gambar 2. Adapun parameter – parameter yang digunakan dalam penelitian ini sebagai berikut :
1. Ukuran populasi = 1000 individu
2. Metode Seleksi = *roulette wheel*
3. Metode Mutasi = Exchange, Insertion dan Inversion
4. Probabilitas mutasi = 0,2

Untuk data yang digunakan dalam penelitin ini menggunakan datasheet TSPLib [14] yang dapat dilihat pada tabel 1. Penggunaan datasheet dari TSPLib karena data tersebut telah memiliki nilai optimum sebagai bahan perbandingan hasil penelitian.

Tabel 1. Datasheet TSPLib

Kasus TSP	Jumlah kota	Nilai Optimum
Berlin52	52	7542
Ch150	150	6528
Pa561	561	2763
Rat783	783	8806
Fl1400	1400	20127
Eil51	51	426
Eil76	76	538
KroA100	100	21282
Lin105	105	14379
Rat195	195	2323
KroB150	150	26130
Gil262	262	2378
Eil101	101	629

Populasi awal dalam penelitian ini yaitu kumpulan urutan perjalanan dari kota ke kota tepat satu kali. Kota – kota dikodekan dalam bentuk bilang desimal dengan rentang [0,gen] dengan gen adalah jumlah kota. Langkah – langkah dalam pembentukan populasi awal adalah sebagai berikut:
1. Menentukan jumlah individu yang dalam suatu populasi (pop) dan jumlah kota anggota individu dalam populasi (gen).
2. Menyiapkan sebuah array populasi dengan ukuran pop x gen.
3. Memilih sebuah angka dengan cara mengacak [0, gen].
4. Memeriksa angka tersebut sudah ada pada individu yang dibentuk.
5. Jika angka yang dipilih sudah ada, mengulangi langkah 3 dan 4.
6. Menyimpan angka yang dipilih pada array populasi.
7. Memeriksa jumlah gen pada individu, jika jumlah gen < jumlah kota maka mengulangi langkah 3 – 6.
8. Memeriksa jumlah individu, jika jumlah individu < populasi maka mengulangi langkah 3 – 7.



Gambar 2. Algoritme Genetika dengan 2 tahap mutasi

Setelah populasi dibangkitkan maka individu – individu akan diseleksi berdasarkan nilai fitness dari masing – masing fitness. Individu yang terpilih merupakan calon induk pada operasi selanjutnya. Operator seleksi yang digunakan dalam penelitian ini yaitu roda rolet (*roulette wheel*). Langkah – langkah dalam menyeleksi induk dengan menggunakan metode roda rolet sebagai berikut:

1. Menghitung nilai fitness dari masing – masing individu dari populasi awal dengan menggunakan persamaan (5).
2. Menghitung fitness komulatif menggunakan persamaan sebagai berikut :

$$f_k = \sum_{i=0}^{i \to n} f[i] \qquad (6)$$

Dengan
f_k = *fitness* komulatif
i = individu pada populasi
n = jumlah individu pada populasi
$f[i]$ = *fitness* individu

3. Menghitung probabilitas fitness (P[i]) dengan menggunakan persamaan sebagai berikut :

$$P[i] = \frac{f[i]}{f_k} \qquad (7)$$

Dengan :
P[i] = Probabilitas *fitness*
f [i] = *fitness* individu
f_k = *fitness* komulatif

4. Menghitung komulatif probabilitas fitness (C[n]) dengan persamaan sebagai berikut :

$$C[n] = \sum_{i=0}^{i \to n} P[i] \qquad (8)$$

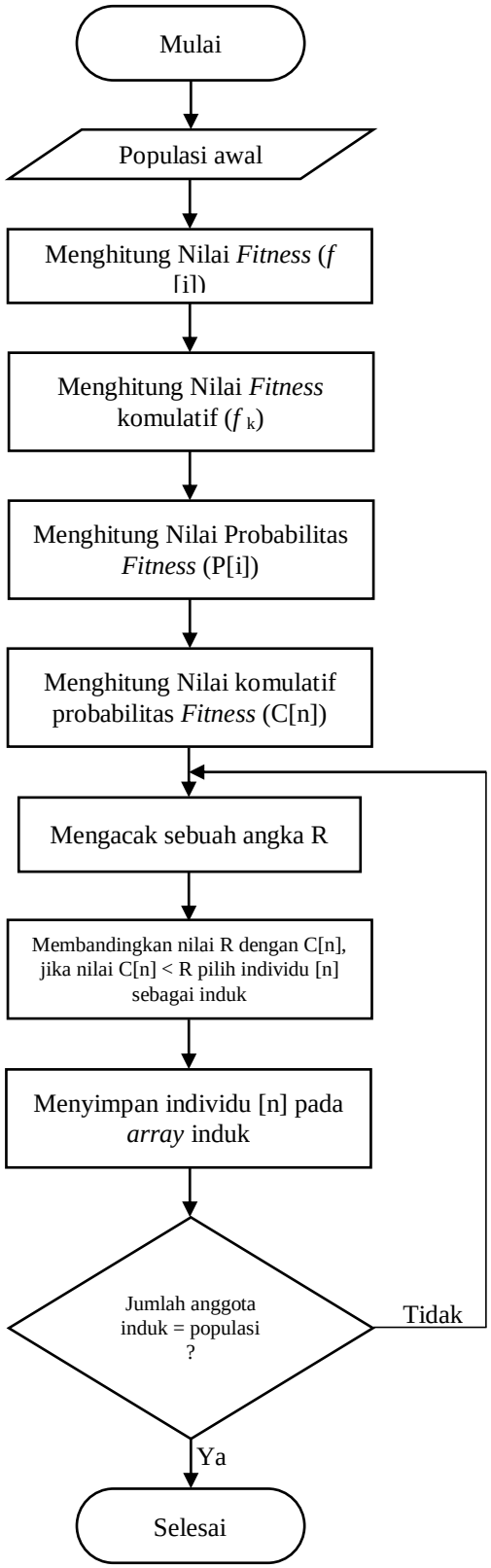
Dengan :
C[n] = Komulatif probabilitas *fitness*.
P[i] = probabilitas *fitness*

5. Menghasilkan sebuah angka acak R [0,1].
6. Membandingkan nilai R dengan nilai C[n], jika nilai C[n] < R maka individu yang terpilih sebagai induk adalah individu[n] pada populasi. Dan simpan individu terpilih ke array induk.
7. Mengulangi langkah 5 dan 6 sampai ukuran array induk = array populasi.

Alur proses dari seleksi induk penelitian ini dapat dilihat pada Gambar 3

Setelah melalui proses seleksi maka akan dicari satu individu yang memiliki nilai fitness yang paling baik. Dalam penelitian ini

nilai fitness terbaik adalah nilai fitness yang paling kecil di dalam populasi. Individu ini akan digandakan sebanyak ukuran populasi dan akan masuk pada proses mutasi.



Gambar 3. *Flowchart* Seleksi

Untuk proses mutasi dalam penelitian ini akan digunakan 2 tahap mutasi. Tahap pertama masing – masing inidividu dalam populasi akan dimutasi dengan menggunakan metode mutasi Exchange dan mutasi Insertion. Individu terbaik dari 2 mutasi akan saling dibandingkan yang memiliki nilai fitness terkecil akan di mutasi kembali dengan metode *Inversion* pada tahap kedua:

4. HASIL DAN PEMBAHASAN

Pengujian dari Algoritme genetika dengan 2 tahap mutasi dengan menguji setiap kasus TSP sebanyak 30 kali percobaan dan menggunakan nilai maximum iterasi/generation yang berbeda pada tiap kota dengan ketentuan sebagai berikut :

1. Untuk jumlah kota: x <= 100, max generation = 100 iterasi.
2. Untuk jumlah kota : 100 < x <= 500, max generation =250 iterasi.
3. Untuk jumlah kota : 500 < x <= 1000, max generation=500 iterasi.
4. Untuk jumlah kota : 1000 < x, max generation =1000 iterasi.

Pengujian dilakukan dengan menghitung nilai error fitness dengan menggunkan rumus sebagi berikut :

$$Error = \frac{Nilai\ 1 - Nilai\ 2}{Nilai\ 2} x100\% \quad (9)$$

Dengan :
Nilai 1 = Nilai Fitness Terbaik
Nilai 2 = Nilai Optimal

Hasil pengujian algoritme genetika dengan 2 tahap mutasi dapat dilihat pada tabel 2. Dan sebagai pembanding hasil penelitian menggunakan hasil penelitian dari [4] yang dapat dilihat pada tabel 3. Penelitian dari [4] menggunakan algoritme genetika dengan menggunakan operasi *crossover*.

Dari tabel 2 dan 3 dapat dilihat, penelitian dengan 2 tahap mutasi menghasilkan nilai persentase erorr fitness yang lebih besar terhadap nilai optimum dari masing – masing kasus kota yang diujikan dibandingkan dengan penelitian dengan menggunakan operasi *crossover* (tabel 3) pada nilai minimum error kecuali pada kasus eil75 penelitian dengan 2 tahap mutasi menghasilkan error yang lebih kecil. Untuk nilai maximum error fitness, penelitian dengan menggunakan 2 tahap mutasi

menghasilkan nilai error yang lebih kecil dari penelitian yang menggunakan operasi *crossover*. Penyebab nilai error yang kecil pada maximum error penelitian ini karena karaketeristik dari operasi mutasi yang tidak melakukan perubahan besar terhadap susunan gen dalam individu, sehingga susunan gen pada individu terbaik yang didapatkan dalam satu iterasi cenderung dipertahankan pada iterasi berikutnya.

Tabel 2. Hasil pengujian metode mutasi 2 tahap

Kasus		Fitness	Persentase (%)
eil51	min	433.74	1.82
	max	444.27	4.29
	rata-rata	439.36	3.14
eil76	min	546.25	1.53
	max	580.47	7.89
	rata-rata	564.92	5.00
kroA100	min	21919.80	3.00
	max	23741.30	11.56
	rata-rata	22760.06	6.95
lin105	min	14754.40	2.61
	max	15848.40	10.22
	rata-rata	15474.78	7.62
rat195	min	2470.29	6.34
	max	2553.25	9.91
	rata-rata	2527.54	8.81
kroB150	min	27283.10	4.41
	max	29133.80	11.50
	rata-rata	27834.22	6.52
gil262	min	2553.21	7.37
	max	2634.97	10.81
	rata-rata	2589.98	8.91
eil101	min	668.96	6.35
	max	674.49	7.23
	rata-rata	671.85	6.81

Tabel 3. Hasil pengujian dengan operator *crossover*-mutasi

Kasus		Fitness	Persentase (%)
Eil51	min	430,2	0,9859
	max	753,7	76,924
Eil76	min	552,3	2,6579
	max	1078,5	100,46
kroA100	min	21294,4	0,0582
	max	40132,6	88,575
Lin105	min	14405,7	0,1856
	max	30500	112,11
Rat195	min	2364,3	1,7778
	max	3986,2	71,597
kroB150	min	26675,9	2,0891
	max	49305,4	88,692
Gil262	min	2506,7	5,4121
	max	3799,5	59,777
Eil101	min	644,5	2,4642
	max	998,3	58,712

5. KESIMPULAN

Dari hasil dan pembahasan penyelesaian masalah TSP dengan Algoritme Genetika dengan menggunakan metode mutasi 2 tahap sudah dapat menyelesaikan permasalahan TSP dengan baik dan layak diimplementasikan. Hasil pengujian terhadap kasus – kasus TSP yang ada pada TSPlib memberikan hasil error rata-rata <10%.

6. DAFTAR PUSTAKA

[1] Rizal, J., "Optimasi Pada Traveling Salesman Problem (TSP) dengan Pendekatan Simulasi Annealing", Jurusan Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Bengkulu, Indonesia, 2007.

[2] Ma, Fangfang dan Li, 2009, "*An algorithm in solving the TSP based on the improved Genetic Algorithm* ", Fundamental Department, Shandong University of Science and Technology, Tai'an, China, The 1st International Conference on Information Science and Engineering (ICISE).

[3] Sutojo, T. dkk., " Kecerdasan Buatan", Penerbit Andi, Yogyakarta, 2010.

[4] Kusum dan Hadush, "*Combined Mutation Operators of Genetic Algorithm for the Travelling Salesman problem*", International Journal of Combinatorial Optimization Problems and Informatics, Vol. 2, No.3, Sep-Dec 2011.

[5] Yingying Yu, dkk., 2011, "*A New Design of Genetic Algorithm for Solving TSP*", Transportation Management College Dalian Maritime University Dalian, China, Fourth International Joint Conference on Computational Sciences and Optimization.

[6] Abramson, D., andAbela, 1992, A ParallelGenetic AlgoritmFor Solving The School Timetabling Problem, 15 Australian Computer Science Conference (ACS-15), 29 – 31 january, pp – 11

[7] Bambrick ., 1997, Lecturer Timetabling Using Genetic Algoritm, Thesis, University of Queensland.

[8] Mahmudy, WF, Marian, RM & Luong, LHS 2014, 'Hybrid genetic algorithms for part type selection and machine loading problems with alternative production plans in flexible manufacturing system', *ECTI Transactions on Computer and Information Technology (ECTI-CIT)*, vol. 8, no. 1, pp. 80-93.

[9] Rahman Erama dan Retantyo Wardoyo, 2013, Modifikasi Algoritma Genetika Untuk Penyelesaian Permasalahan Penjadwalan Pelajaran Sekolah, Prosiding Seminar Nasional Ilmu Komputer 2013, Indonesia Computer and Instrumentation Support Society (indoCEIS), pp-47

[10] Riska Hardini Purnamasari , Implementasi Algoritme Genetika Untuk Pencarian Rute Minimum Dalam Travelling Salesman Problem, Universitas Komputer Indonesia, 2009.

[11] Alexander Schrijver, *"Geometric Algorithms And Combinatorial Optimization"*, Springer-Verlag, New York,1988.

[12] Kusumadewi, S dan Purnomo, H., "Aplikasi Logika Fuzzy". Graha Ilmu Yogjakarta, 2010.

[13] Golberg, D. E., "Genetic *Algorithm In Search, Optimization And Machine Learning*", New York: Addison-Wesley. 1989.

[14] Reinelt, G. 2004, "*TSPLIB is a library of sample instances for the TSP (and related problems) from various sources and of various types", http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/STSP.html*, diakses tanggal 2 September 2015.