PRISMA 3 (2020): 1-7
PRISMA, PROSIDING SEMINAR NASIONAL MATEMATIKA
https://journal.unnes.ac.id/sju/index.php/prisma/
ISSN 2613-9189

Analisis Pengaruh Operator Genetik pada Algoritma Genetika dan Penerapannya pada *Traveling Salesman Problem* (TSP)

Nuzulul Khairu Nissa[a,1], Farikhin[b,2], Bayu Surarso[c,3]

[a,b,c] *Department of Mathematics, Diponegoro University, Tembalang, Semarang 50275, Indonesia*

[1,2,3] *Alamat Surel: nuzululkha@gmail.com, farikhin.math.undip@gmail.com, bayusurarso@yahoo.com*

Abstrak

Algoritma genetika merupakan suatu algoritma pencarian metaheuristik yang berdasar pada mekanisme seleksi alam dan operasi genetika guna diperolehnya suatu solusi. Solusi yang dihasilkan, ditentukan berdasarkan nilai parameter dan operator yang digunakan. Penentuan cara kerja dari masing-masing operator yaitu operator seleksi, *crossover* dan mutasi, dapat diketahui dengan menggunakan teori *Schemata*. Selain itu, jika diasumsikan bahwa operator *crossover* diabaikan, dengan maksud akan lebih ditekankan pada operator mutasi maka probabilitas dihasilkannya suatu solusi, akan meningkat. Salah satu permasalahan yang dapat diselesaikan dengan menggunakan algoritma genetika yaitu *Traveling Salesman Problem* (TSP). TSP merupakan suatu permasalahan optimasi, guna ditemukannya sikel Hamilton yang memiliki bobot minimum pada sebuah graf terhubung. Penelitian ini membahas penerapan algoritma genetika untuk menyelesaikan TSP pada kasus data ulysses16.tsp. Berdasarkan perhitungan dan pengujian, diperoleh nilai parameter yang menghasilkan total jarak tempuh minimal sebesar 65.169, dengan: jumlah generasi sebesar 300, jumlah populasi sebesar 120, probabilitas *crossover* sebesar 0,7 dan probabilitas mutasi sebesar 0,3.

Kata kunci:

Algoritma Genetika, *Traveling Salesman Problem* (TSP)

1. Pendahuluan

Algoritma genetika merupakan suatu algoritma pencarian yang berbasis pada mekanisme seleksi alam dan genetika. Algoritma genetika memiliki 3 operator genetik utama yaitu operator seleksi, *crossover* dan mutasi, yang mana masing-masing operator tersebut memiliki perannya masing-masing guna dihasilkannya suatu solusi yang mendekati optimal (Arkeman *et al*., 2012)

Traveling Salesman Problem (TSP) merupakan salah satu permasalahan optimasi yang dapat diselesaikan dengan menggunakan algoritma genetika. Tujuan utama dari TSP yaitu, untuk mencari rute terpendek dari suatu tempat ke tempat lain yang akan dituju oleh seorang *salesman* dengan syarat setiap tempat hanya akan dikunjungi satu kali dan harus kembali ke tempat asal (Applegate *et al*., 2006). Dalam teori Graf, tujuan dari TSP adalah untuk menemukan sirkuit *Hamilton* yang memiliki bobot minimum pada sebuah graf terhubung (Goldberg, 1989). Pada penelitian ini, akan dibahas terkait dengan analisis cara kerja dari masing-masing operator algoritma genetika dan penerapannya dalam menyelesaikan TSP pada data *ulysses16.tsp* yang diperoleh dari *Traveling Salesman Problem Library* (TSPLIB).

2. Pembahasan

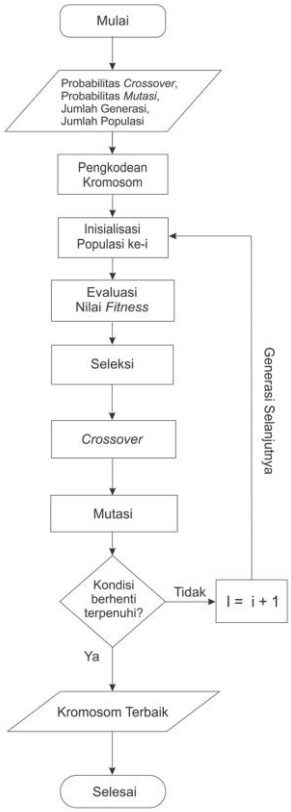
2.1 Algoritma Genetika

Algoritma genetika merupakan algoritma pencarian metaheuristik yang berbasis pada mekanisme seleksi alam dan genetika. Algoritma genetika bekerja menciptakan banyak solusi secara acak dengan menggunakan aturan probabilistik. Algoritma genetika menciptakan populasi baru melalui iterasi secara

To cite this article:
Nissa, N.K., Farikhin, & Surarso, B. (2020). Analisis Pengaruh Operator Genetik pada Algoritma Genetika dan Penerapannya pada *Traveling Salesman Problem* (TSP). *PRISMA, Prosiding Seminar Nasional Matematika 3*, 1-7

terus menerus terhadap populasi awal sampai didapatkannya populasi dengan harapan solusi tersebut semakin dekat dengan solusi optimal permasalahan yang ada (Suyanto, 2005).

Penjelasan proses dari algoritma genetika diilustrasikan pada Gambar 1.



Gambar 1. Proses algoritma genetika

Penjelasan langkah-langkah dari algoritma genetika berdasarkan ilustrasi pada Gambar 1. disajikan pada Tabel 1.

Tabel 1. Langkah-langkah penjelasan proses algoritma genetika

Proses	Penjelasan	
(Input)	Menentukan nilai parameter:	
	Probabilitas *crossover*	Jumlah generasi
	Probabilitas mutasi	Jumlah populasi
Langkah 1	Pengkodean kromosom, yaitu proses merepresentasikan gen-gen dari setiap kromosom.	
Langkah 2	Inisialisasi populasi ke-*i*, yaitu proses pembangkitan secara acak populasi kromosom untuk generasi ke-*i*.	
Langkah 3	Evaluasi nilai *fitness*, yaitu perhitungan masing-masing nilai dari fungsi objektif, untuk kemudian dicari nilai *fitness* yang menunjukkan kualitas dari masing-masing kromosom.	
Langkah 4	Seleksi, yaitu proses pemilihan kromosom untuk dijadikan *parents* yang selanjutnya akan melalui proses *crossover*.	
Langkah 5	*Crossover*, yaitu proses penyilangan kromosom sehingga membentuk kromosom baru (*offspring*) yang harapannya lebih baik dari pada kualitas *parent*-nya.	
Langkah 6	Mutasi, yaitu proses pengubahan nilai dari satu atau beberapa gen pada kromosom.	
Jika kondisi iterasi berhenti terpenuhi, diperoleh kromosom terbaik.		
Jika kondisi iterasi berhenti tidak terpenuhi, dilakukan iterasi ke-(i+1)		
(Output)	Kromosom terbaik	

2.2 Analisis Cara Kerja Algoritma Genetika dengan Teori Schemata

Schema merupakan rancangan umum dari beberapa kromosom yang memiliki kemiripan dari nilai *fitness* maupun strukturnya. Dalam analisis matematis cara kerja algoritma genetika dengan teori *schemata* yaitu dimisalkan terdapat sebuah *schema* dan diharapkan terjadi peningkatan nilai *fitness* pada *schema* tersebut seiring dengan meningkatnya jumlah generasi. Peningkatan nilai *fitness* ini dipengaruhi oleh operator genetik diantaranya, operator seleksi, *crossover* dan mutasi (Arkeman *et al.*, 2012).

Teorema 1 (Arkeman *et al.*, 2012) *Misalkan terdapat sebuah schema yang diberi nama H dan* $\varepsilon(H, k+1)$ *merupakan peluang terpilihnya kromosom dalam schema H dari suatu populasi*$P(k)$*, berlaku:*

$$\varepsilon(H,k+1) \geq \left(1-\rho_c \frac{d(H)}{L-1}\right)(1-\rho_m)^{o(H)}\left(\frac{f(H,k)}{\bar{F}(k)}\right)e(H,k)$$

Berdasarkan pada Teorema 1, dapat disimpulkan bahwa pengaruh dari masing-masing operator genetik pada algoritma genetika adalah sebagai berikut (Arkeman *et al.*, 2012):

- $\frac{f(H,k)}{\bar{F}(k)}$, mengindikasikan pengaruh nilai *fitness* rata-rata dari sebuah *schema H.*
- $1-\rho_c \frac{d(H)}{L-1}$, mengindikasikan pengaruh operator *crossover* .
- $(1-\rho_m)^{o(H)}$, mengindikasikan pengaruh operator mutasi.

Dari pengaruh ketiga operator diatas, dapat disimpulkan bahwa sebuah *schema* dikatakan cukup baik, jika *schema* tersebut berordo kecil (*low order*), ber-*distance* rendah (*short*) dan memiliki rata-rata nilai *fitness* yang lebih baik dibandingkan rata-rata nilai *fitness* populasi (Arkeman *et al.*, 2012).

2.3 Konvergensi Algoritma Genetika

Kromosom-kromosom yang telah melalui proses mutasi, secara intuitif merupakan kromosom-kromosom yang menghasilkan suatu solusi yang mendekati optimal. Oleh karenanya pada bagian ini, diasumsikan proses *crossover* diabaikan dan akan lebih diprioritaskan pengaruh dari proses mutasi. Probabilitas ditemukannya solusi permasalahan akan semakin besar jika operator mutasi semakin berperan (Sharapov & Lapshin, 2006).

Untuk membuktikan pernyataan bahwa setelah proses mutasi akan terjadi peningkatan jumlah elemen berupa elemen solusi di populasi, maka langkah pertama yang harus dilakukan yaitu mencari probabilitas bahwa mutasi tidak akan meningkatkan jumlah elemen solusi di populasi yang berdasar pada Lemma 1.

Lemma 1 (Sharapov & Lapshin, 2006) *Misal* P_{mt} *merupakan probabilitas dari mutasi dengan* $P_{mt} \leq 0{,}5$ *dan* $s^* \in S$ *merupakan individu yang terseleksi.*

$$P\{M\} = \frac{1}{C_m^k} P_{mt}^k (1-P_{mt})^{m-k}$$

dengan M merupakan keadaan ketika s termutasi pada s^* *atau dinotasikan dengan* $(s \to s^*)$*. Maka* $P\{M\}$ *merupakan probabilitas bahwa s termutasi pada* s^**. Akan dibuktikan* $P\{Q^c\}$ *yang merupakan probabilitas bahwa* $s^* \notin mut(Pop)$ *yaitu:*

$$P\{Q^c\} \leq \left(1-\frac{P_{mt}^m}{C_m^{[m/2]}}\right)^n$$

Untuk selanjutnya, Teorema 2 menunjukkan bahwa algoritma genetika yang mengandung *elitism* akan konvergen pada suatu nilai *fitness* optimal yang dinotasikan dengan f^*. *Elitism* merupakan suatu proses dalam algoritma genetika dimana posisi-posisi kromosom yang memiliki nilai *fitness* yang rendah akan digantikan oleh kromosom-kromosom *elite* yang pada tahap evaluasi nilai *fitness* telah di-*copy* dan disimpan (Sutojo, 2011).

Teorema 2 (Sharapov & Lapshin, 2006) *Dimisalkan suatu algoritma genetika mengandung elitism dengan* $P_{mt} \leq 0{,}5$ jika:

$$S = \left(1-\frac{p_{mt}^m}{C_m^{[m/2]}}\right)^n (2-(1-p_c)^n) < 1$$

maka,

$$Mf(Pop^k) \xrightarrow[k\to\infty]{} f^*$$

(tanpa monoton turun)

2.4 Algoritma Genetika pada Traveling Salesman Problem (TSP)

Pada penelitian ini, data TSP yang digunakan adalah data *ulysses16.tsp* yang diperoleh dari *Traveling Salesman Problem-Library* (TSPLIB).

Berikut ini merupakan langkah-langkah dari proses algoritma genetika untuk menyelesaikan TSP data kasus *ulysses16.tsp*:

Langkah 1. Pengkodean kromosom yang digunakan pada kasus ini adalah *permutation encoding.*

Langkah 2. Inisialisasi populasi atau proses pembangkitan populasi awal yang digunakan pada kasus ini adalah *random generator.*

Langkah 3. Evaluasi nilai *fitness* atau proses perhitungan nilai fungsi objektif dan nilai *fitness* dari masing-masing kromosom menggunakan persamaan (1) dan (2). Langkah selanjutnya yaitu meng-*copy*-kan kromosom-kromosom yang memiliki nilai *fitness* tertinggi sebagai kromosom *elite.*

$$Total\_Jarak[i] = \sum_{i=1}^{n-1} (c_{i,i+1} + c_{n,1}) \quad (1)$$

$$fitness[i] = \frac{1}{Total\_Jarak[i]} \quad (2)$$

Langkah 4. Metode seleksi yang digunakan adalah *roulette wheel selection* dengan *Linear Fitness Ranking* (LFR).

Langkah 4.1 Melakukan perhitungan LFR dengan menggunakan persamaan (3)

$$LFR[i] = (f\left(\frac{r(i)-1}{N-1}\right) minmax_{max} \quad (3)$$

Langkah 4.2 Menghitung total nilai LFR dengan menggunakan persamaan (4)

$$Total\_LFR = \sum_{i=1}^{n} LFR[i] \quad (4)$$

Langkah 4.3 Menghitung nilai LFR kumulatif dengan menggunakan persamaan (5)

$$LFR\_cum[n] = \sum_{i=1}^{n} LFR[i] \quad (5)$$

Langkah 4.4 Menghitung nilai LFR relatif dengan menggunakan persamaan (6)

$$P_{cum}(n) = \frac{LFR\_cum[n]}{Total\_LFR} \quad (6)$$

Langkah 4.5 Memilih *parent* untuk di *crossover*-kan dengan cara: Membangkitkan bilangan *random* R dengan rentang [0,1) sebanyak jumlah kromosom pada populasi. Jika $P_{cum}(n) \leq R[i]$ dan $P_{cum}(n+1) > R[i]$ maka pilih kromosom[*n*] sebagai kandidat *parent* (Sutojo, 2011).

Langkah 5. *Order Crossover*

Langkah 5.1 Menentukan nilai probabilitas *crossover* (ρ_c).

Langkah 5.2 Membangkitkan bilangan *random* sebanyak jumlah populasi, kromosom[i] akan terpilih sebagai *parent* jika $R[i] < \rho_c$.

Langkah 5.3 Melakukan proses *crossover* dengan menggunakan *order crossover.*

Contoh 1. Misal, terdapat dua *parent* yaitu P_1 dan P_2 yang akan melakukan proses *crossover.*

P_1 = [	6	5	15	14	16	3	2	4	8	1	13	12	7	10	9	11	]
P_2 = [	10	7	12	3	2	4	8	1	16	13	14	15	5	6	9	11	]

Bit-bit yang terletak diantara 2 titik potong disalin dengan cara yang sama ke *offspring*,sehingga diperoleh:

O_1 = [	×	×	×	×	×	3	2	4	8	1	13	×	×	×	×	×	]
O_2 = [	×	×	×	×	×	4	8	1	16	13	14	×	×	×	×	×	]

Selanjutnya dimulai dari titik potong kedua dari salah satu *parent*, bit-bit dari *parent* lain disalin dalam urutan yang sama, selanjutnya beberapa bit yang ada dihilangkan. Kemudian, urutan ini diletakkan di O_1 dimulai dari titik potong yang kedua, sehingga diperoleh *offspring*[1] dan analogi dengan proses membangun *offspring*[1], maka diperoleh *offspring*[2].

O_1 = [	10	7	12	16	14	3	2	4	8	1	13	15	5	6	9	11	]
O_2 = [	6	5	15	3	2	4	8	1	16	13	14	12	7	10	9	11	]

Langkah 6. Proses Mutasi

Langkah 6.1 Menentukan nilai probabilitas mutasi (ρ_m)

Langkah 6.2 Menghitung panjang gen dalam 1 populasi menggunakan persamaan (7)

Panjang Total Gen = Jumlah gen dalam 1 kromosom x jumlah kromosom (7)

Langkah 6.3 Menentukan jumlah gen yang termutasi menggunakan persamaan (8)

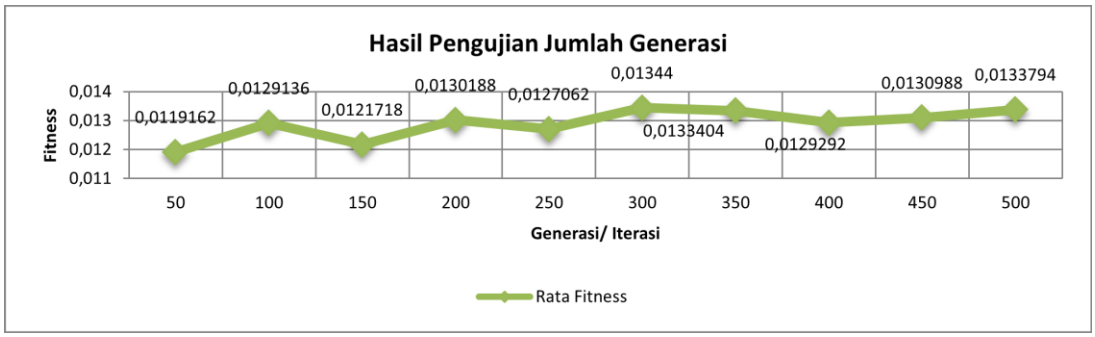
$$\text{Jumlah Gen yang Termutasi} = \text{nilai } \rho_m \times \text{total gen} \quad (8)$$

Langkah 6.4 Membangkitkan bilangan *random* (*integer*) antara 1 hingga panjang total gen untuk menentukan posisi gen yang akan termutasi.

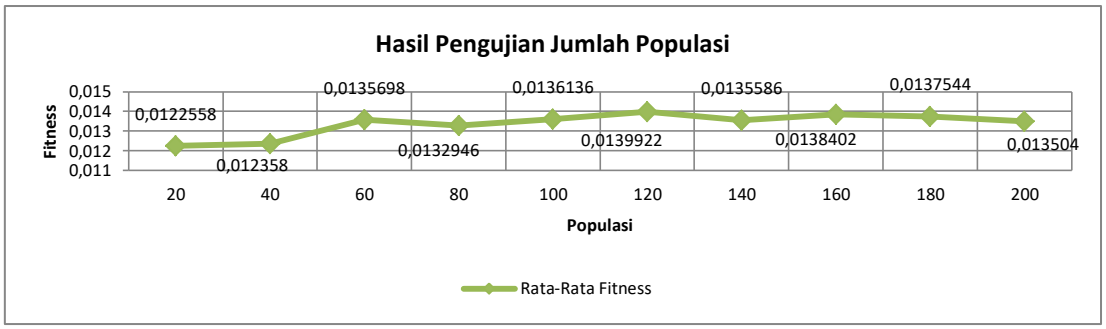
Langkah 7. Proses *Elitism* yaitu proses penggantian posisi-posisi kromosom yang memiliki nilai *fitness* rendah yang digantikan dengan kromosom *elite* yang pada tahap evaluasi nilai *fitness* telah di-*copy* dan disimpan (Sharapov & Lapshin, 2006).

2.5 Pengujian Parameter Algoritma Genetika

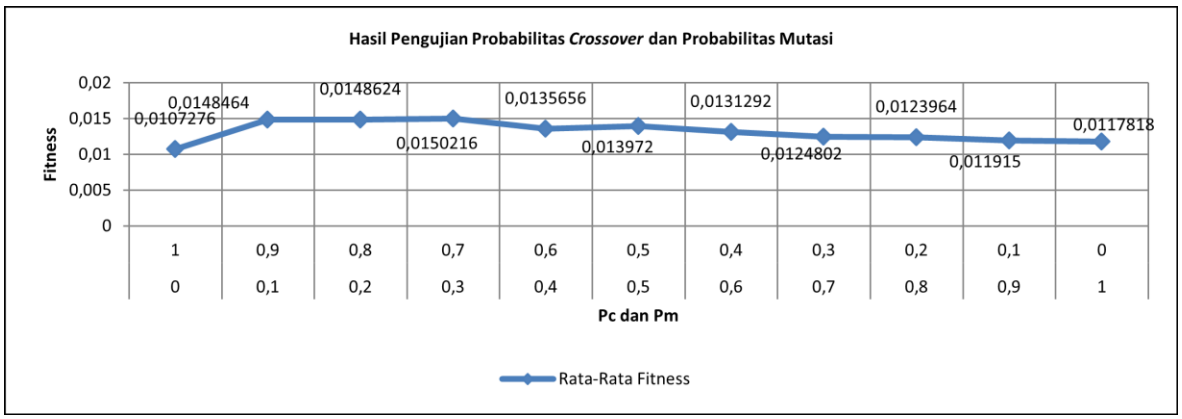
Pengujian yang dilakukan, yaitu dengan mengkombinasikan beberapa parameter algoritma genetika berupa jumlah generasi, jumlah populasi, probabilitas *crossover* (ρ_c) dan probabilitas mutasi (ρ_m). Pengujian yang dilakukan akan memberikan hasil berupa nilai-nilai parameter yang akan digunakan untuk memperoleh solusi yang mendekati optimal. Hasil pengujian parameter-parameter algoritma genetika tersebut disajikan pada Gambar 2, Gambar 3 dan Gambar 4.



Gambar 2. Hasil pengujian jumlah generasi

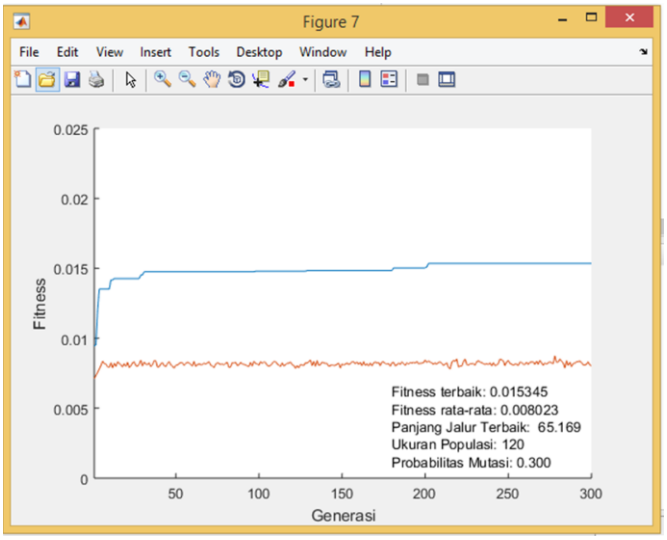


Gambar 3. Hasil pengujian jumlah populasi



Gambar 4. Hasil pengujian probabilitas *crossover* dan probabilitas mutasi

Berdasarkan hasil pengujian, nilai parameter yang dihasilkan yaitu, jumlah generasi sebesar 300, jumlah populasi sebesar 120, probabilitas *crossover* sebesar 0,7 dan probabilitas mutasi sebesar 0,3. Selanjutnya, dengan menggunakan Matlab dilakukan penginputan nilai parameter hasil pengujian dan diperoleh hasil berupa rata-rata nilai *fitness* dan jalur terbaik yang disajikan pada Gambar 5 dan Gambar 6.



Gambar 5. Nilai *fitness* berdasarkan parameter hasil pengujian

Gambar 6. Jalur terbaik berdasarkan parameter hasil pengujian

Gambar 5 dan Gambar 6 merupakan hasil *running program* Matlab dengan input parameter terbaik yang diperoleh dari hasil pengujian dan diperoleh nilai *fitness* terbaik sebesar 0,015345, dengan rute jalur terbaik yang dihasilkan sebagai berikut:

$$6 \rightarrow 5 \rightarrow 15 \rightarrow 14 \rightarrow 16 \rightarrow 3 \rightarrow 2 \rightarrow 4 \rightarrow 8 \rightarrow 1 \rightarrow 13 \rightarrow 12 \rightarrow 7 \rightarrow 10 \rightarrow 9 \rightarrow 11$$

Berdasarkan jalur rute terbaik tersebut dapat disimpulkan bahwa, *salesman* tersebut akan mengawali perjalanan dari kota-6, kemudian ke kota-5 dst, sampai ke kota-11 untuk kembali lagi ke kota-6. Panjang jalur terbaik yang diperoleh yaitu sebesar 65,169 dan waktu komputasi sebesar 104,0938.

3. Simpulan

Langkah-langkah proses algoritma genetika untuk menyelesaikan suatu permasalahan khususnya *Traveling Salesman Problem* (TSP) terdiri dari, pengkodean kromosom, inisialisasi populasi, evaluasi nilai *fitness*, proses seleksi, proses *crossover*, proses mutasi dan proses *elitism*. Untuk menganalisis cara kerja dari algoritma genetika dapat diketahui dengan menggunakan teori *Schemata*.

Algoritma genetika akan konvergen pada suatu nilai *fitness* optimal yang dinotasikan dengan f^*dengan asumsi bahwa algoritma tersebut mengandung *Elitism* serta diasumsikan bahwa proses *crossover* akan diabaikan dan lebih diprioritaskan pada proses mutasi.

Pada penelitian ini, algoritma genetika dipelajari untuk menyelesaikan *Traveling Salesman Problem* (TSP) pada kasus data *ulysses16.tsp*. Dari hasil perhitungan diperoleh jarak tempuh minimal sebesar

65.169 dengan rute [6 5 15 14 16 3 2 4 8 1 13 12 7 10 9 11]. Parameter *input* yang digunakan untuk menyelesaikan kasus tersebut diperoleh berdasarkan hasil proses pengujian yaitu: dengan jumlah generasi sebesar 300, jumlah populasi sebesar 120, probabilitas *crossover* sebesar 0,7 dan probabilitas mutasi sebesar 0.3.

Daftar Pustaka

Arkeman, Y., Seminar, K. B., & Gunawan, H. (2012). *Algoritma Genetika, Teori dan Aplikasinya untuk Bisnis dan Industri*. Bogor: PT Penerbit IPB Press.

David L. Applegate, Robert E. Bixby, Vasek Chvatal, & William J. Cook. (2006). *The Traveling Salesman Problem - A Computational Study*. New Jersey: Princeton University Press.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Massachusett: Addison-Wesley.

Sharapov, R.R., & Laphshin, A.V. (2006). Convergence of Genetic Algorithm. *Pattern Recognition and Image Analysis, 16*, 392-397.

Sutojo, T. (2011). *Kecerdasan Buatan.* Yogyakarta: ANDI.

Suyanto. (2005). *Algoritma Genetika dalam MATLAB*. Yogyakarta: Andi Offset.