Jurnal Komputer dan Teknologi Informasi

Vol. 3, No. 2, Juli 2025, pp. 73~79
E-ISSN: 2986-7592, DOI: 10.26714/jkti.v3i2.16251

Optimalisasi Rute *Traveling Salesman* Menggunakan Algoritma Genetika

Yesi Fitria Sari[1], Dr. Muhammad Munsarif, S.Kom., M.Kom.[2]
[1]Program Studi Informatika, Fakultas Teknik Universitas Muhammadiyah Semarang, Semarang, Indonesia
[2]Program Studi Informatika, Fakultas Teknik Universitas Muhammadiyah Semarang, Semarang, Indonesia

Info Artikel

Riwayat Artikel:

Diterima 10 Mei 2025
Perbaikan 13 Juni 2025
Disetujui 28 Juli 2025

Keywords:

Traveling Salesman Problem
Algoritma Genetika
Optimasi Rute
Natural Selection
Pemodelan Logistik

ABSTRAK

Permasalahan Traveling Salesman (TSP) berfokus pada pencarian jalur terpendek yang memungkinkan seorang agen perjalanan mengunjungi setiap kota satu kali saja, kemudian kembali ke lokasi awal.. Penelitian ini menerapkan Algoritma Genetika (AG) sebagai metode optimasi untuk mengatasi TSP, dengan memanfaatkan dataset koordinat lokasi sebagai representasi titik-titik yang harus dikunjungi. Proses dalam algoritma ini mengikuti konsep evolusi biologis, seperti seleksi, pewarisan, dan perubahan acak, guna menemukan solusi terbaik dengan jarak perjalanan seminimal mungkin. Penelitian ini juga menganalisis pengaruh parameter AG, seperti ukuran populasi, tingkat crossover, dan tingkat mutasi, terhadap kualitas solusi yang dihasilkan. Hasil eksperimen menunjukkan bahwa dengan pengaturan parameter yang tepat, AG dapat menghasilkan rute yang mendekati optimal secara efisien dan konsisten.

ABSTRACT

The Traveling Salesman Problem (TSP) focuses on determining the shortest possible route that allows a traveling agent to visit each city exactly once before returning to the starting point. This study applies the Genetic Algorithm (GA) as an optimization method to solve the TSP by utilizing a dataset of location coordinates representing the points that must be visited. The algorithm operates based on biological evolutionary principles such as selection, inheritance, and random mutation, in order to discover the most efficient route with minimal travel distance. This research also examines the impact of GA parameters—such as population size, crossover rate, and mutation rate—on the quality of the resulting solutions. Experimental results indicate that with proper parameter settings, GA can generate near-optimal routes efficiently and consistently.

Penulis Korespondensi:

Penulis Pertama
Program Studi Informatika, Fakultas Teknik Universitas Muhammadiyah Semarang
Alamat: Gedung FT-MIPA Lt. 7, Ruang 707, Jl.Kedungmundu Raya No.18, Semarang 50273, Indonesia
Email: penulispertama@unimus.ac.id

1. PENDAHULUAN

Masalah Traveling Salesman (TSP) telah menjadi subjek penelitian utama dalam ilmu komputer dan matematika kombinatorik selama beberapa dekade. TSP, sebagai salah satu permasalahan optimasi klasik, menuntut pencarian rute terpendek yang memungkinkan seorang pengembara atau salesman untuk mengunjungi setiap kota hanya sekali, sebelum akhirnya kembali ke titik awal. Esensi dari TSP adalah

minimasi jarak tempuh secara keseluruhan, yang menjadikannya relevan dalam berbagai aplikasi praktis[1]. Misalnya, TSP dapat diterapkan pada logistik untuk merancang rute distribusi barang yang efisien, pada perencanaan transportasi untuk menentukan jalur kendaraan yang optimal, serta dalam berbagai aplikasi lain yang membutuhkan perencanaan rute yang hemat waktu dan biaya.

Seiring dengan bertambahnya jumlah kota yang harus dikunjungi, kompleksitas TSP meningkat secara eksponensial. Pencarian solusi optimal dalam TSP menjadi tantangan besar karena jumlah kombinasi rute yang mungkin bertambah drastis sejalan dengan jumlah titik tujuan [2]. Kombinatorialitas yang tinggi ini membuat pencarian solusi optimal secara langsung tidak efisien dan sering kali tidak praktis jika mengandalkan metode pencarian eksplisit atau algoritma eksak, terutama pada skala yang besar. Oleh karena itu, diperlukan pendekatan berbasis heuristik atau metaheuristik yang mampu mengeksplorasi ruang solusi secara efektif dan menghasilkan solusi yang mendekati optimal dalam waktu yang lebih singkat.

Algoritma Genetika (AG) adalah salah satu metode yang banyak digunakan untuk menyelesaikan masalah optimasi kompleks, termasuk TSP. AG merupakan algoritma metaheuristik yang terinspirasi dari teori evolusi biologis, di mana solusi-solusi potensial direpresentasikan sebagai "individu" dalam suatu "populasi" [3]. Melalui serangkaian proses seleksi, *crossover*, dan mutasi, AG memungkinkan setiap generasi populasi berevolusi, dengan tujuan memperbaiki kualitas solusi dari generasi ke generasi. Dalam proses seleksi, individu-individu dengan nilai fitness terbaik, yang dalam konteks TSP berarti memiliki jarak tempuh yang paling pendek, diprioritaskan untuk menghasilkan keturunan [4]. *Crossover* atau rekombinasi memungkinkan penggabungan karakteristik terbaik dari dua solusi, sedangkan mutasi memperkenalkan variasi acak yang mencegah populasi terkunci pada solusi suboptimal.

Penelitian ini bertujuan untuk mengimplementasikan Algoritma Genetika dalam menyelesaikan TSP dengan menggunakan dataset yang terdiri dari koordinat lokasi sebagai representasi titik-titik pengiriman yang harus dikunjungi. Kami memanfaatkan dataset ini untuk menguji performa AG dalam menemukan rute optimal. Selain itu, kami juga menganalisis bagaimana parameter-parameter dalam AG, seperti ukuran populasi, tingkat *crossover*, dan tingkat mutasi, mempengaruhi kualitas solusi yang dihasilkan. Dengan pengaturan parameter yang tepat, kami berharap AG tidak hanya mampu menghasilkan rute yang mendekati optimal, tetapi juga memberikan pendekatan yang bermanfaat dan aplikatif bagi permasalahan rute dalam konteks nyata, khususnya dalam sektor logistik dan distribusi.

2. METODE

2.1 Algroritma Genetika

Algoritma Genetika (AG) merupakan teknik optimasi yang banyak digunakan dalam menyelesaikan masalah-masalah kompleks, termasuk *Traveling Salesman Problem* (TSP). Algoritma ini bekerja dengan membentuk populasi awal yang terdiri dari sejumlah solusi acak. Setiap solusi tersebut merepresentasikan satu kemungkinan rute perjalanan yang akan dievaluasi berdasarkan total jarak tempuh yang dihasilkan. Dalam konteks TSP, semakin pendek jarak tempuh, semakin baik kualitas atau nilai fitness dari solusi tersebut. AG menggunakan pendekatan iteratif untuk melakukan seleksi terhadap solusi-solusi terbaik di setiap generasi [5]. Solusi terpilih kemudian direproduksi melalui mekanisme *crossover* untuk membentuk generasi baru yang diharapkan lebih mendekati solusi optimal. Proses ini diulangi hingga ditemukan solusi optimal atau hingga mencapai jumlah iterasi yang telah ditentukan.

Untuk mencegah terkuncinya populasi pada solusi optimal lokal, AG menerapkan mutasi pada individu dalam populasi. Proses mutasi berperan dalam menjaga variasi individu dalam populasi, yang membantu algoritma menjelajahi ruang solusi secara lebih menyeluruh. Mutasi yang diterapkan dengan probabilitas tertentu dapat meningkatkan peluang untuk menemukan solusi yang lebih baik, terutama pada masalah yang kompleks seperti TSP [6].

2.2 Tahapan Algoritma

Algoritma Genetika terdiri dari beberapa tahapan utama yang secara bertahap memperbaiki populasi solusi dalam upaya mencapai rute optimal. Berikut adalah penjelasan lebih rinci mengenai setiap tahap yang dilakukan oleh Algoritma Genetika dalam konteks penyelesaian TSP [7]:

1. Inisialisasi: Pada tahap pertama, algoritma membentuk populasi awal dengan menghasilkan sejumlah solusi acak. Setiap solusi dalam populasi awal ini menggambarkan urutan kunjungan antar kota yang berbeda-beda. Dengan memiliki variasi yang cukup dalam populasi awal, algoritma meningkatkan peluang untuk menemukan solusi optimal, karena kombinasi rute yang beragam dieksplorasi sejak awal.

2. Seleksi: Setelah populasi awal terbentuk, algoritma melakukan seleksi untuk memilih individu atau solusi terbaik yang akan direproduksi. Dalam penelitian ini, digunakan metode tournament selection, di mana sekelompok individu dari populasi dipilih secara acak untuk berkompetisi. Individu dengan nilai fitness tertinggi dalam kelompok tersebut akan dipilih sebagai kandidat untuk proses *crossover*. Metode seleksi ini memastikan bahwa solusi terbaik memiliki peluang lebih besar untuk diwariskan pada generasi berikutnya, sehingga mendorong evolusi populasi menuju solusi optimal.
3. *Crossover*: Pada tahap crossover, dua solusi terbaik yang terpilih sebagai "orang tua" akan menghasilkan dua solusi baru atau "anak" melalui metode *Partially Mapped Crossover* (PMX). Metode PMX menjaga sebagian urutan kunjungan kota dari kedua orang tua agar tetap ada pada anak-anak yang dihasilkan, sehingga karakteristik rute terbaik tetap terjaga. Dengan menggabungkan karakteristik dari dua solusi yang memiliki nilai fitness tinggi, diharapkan keturunan yang dihasilkan mendekati atau bahkan mencapai solusi optimal. Tahap *crossover* ini merupakan kunci dalam Algoritma Genetika, karena memungkinkan terjadinya perpaduan sifat-sifat terbaik dari solusi yang ada.
4. Mutasi: Setelah proses *crossover*, algoritma menerapkan mutasi pada solusi-solusi baru untuk mempertahankan variasi dalam populasi. Mutasi dilakukan dengan mengganti posisi dua titik kunjungan dalam solusi, yang dikenal sebagai *swap mutation*. Mutasi ini dilakukan dengan probabilitas tertentu, yang memungkinkan algoritma mencegah terjebaknya solusi pada titik optimal lokal. Dengan menjaga keragaman dalam populasi melalui mutasi, algoritma dapat mengeksplorasi ruang solusi yang lebih luas, sehingga meningkatkan peluang untuk menemukan solusi optimal.
5. Evaluasi: Setiap solusi baru yang dihasilkan dari proses *crossover* dan mutasi kemudian dievaluasi berdasarkan total jarak antar titik kunjungan. Algoritma menghitung nilai *fitness* setiap solusi, di mana solusi dengan jarak tempuh terpendek akan memiliki nilai *fitness* yang lebih tinggi. Nilai *fitness* ini menjadi dasar dalam proses seleksi berikutnya, memastikan bahwa solusi dengan performa terbaik akan diutamakan dalam reproduksi generasi berikutnya. Proses evaluasi yang iteratif ini memungkinkan populasi untuk berkembang secara bertahap menuju solusi yang lebih optimal.
6. Generasi Baru: Proses dari tahap inisialisasi hingga evaluasi diulang secara berkesinambungan untuk setiap generasi hingga jumlah generasi yang ditetapkan tercapai atau hingga solusi optimal ditemukan. Dengan setiap generasi baru, kualitas solusi terus meningkat, karena proses seleksi dan mutasi bekerja untuk memperbaiki populasi secara bertahap. Proses ini memberikan kesempatan pada algoritma untuk mempersempit ruang solusi secara bertahap dan mendekati hasil yang optimal.

2.3 Fungsi *Fitness*

Fungsi *fitness* memainkan peran esensial dalam Algoritma Genetika karena fungsi ini menjadi ukuran kualitas solusi berdasarkan total jarak yang ditempuh dalam rute [6]. Semakin pendek jarak tempuh yang dihasilkan oleh sebuah solusi, semakin tinggi nilai *fitness* yang dimilikinya. Dalam konteks TSP, solusi dengan jarak tempuh terpendek diprioritaskan dalam seleksi, sehingga solusi terbaik memiliki peluang lebih besar untuk dikembangkan dalam generasi berikutnya [4]. Dalam penelitian ini, fungsi *fitness* membantu algoritma dalam memilih solusi-solusi unggul yang diharapkan dapat menghasilkan rute optimal seiring dengan bertambahnya generasi.

2.4 Dataset

Dataset yang digunakan dalam penelitian ini berisi data koordinat dua dimensi dari sejumlah titik lokasi yang mewakili kota-kota yang harus dikunjungi dalam konteks *Traveling Salesman Problem* (TSP). Dataset berisi 99 entri, masing-masing menggambarkan satu titik lokasi dengan informasi posisi dua dimensi yang disimpan dalam dua kolom koordinat. Kedua kolom tersebut menyimpan nilai koordinat dalam bentuk titik desimal yang menunjukkan posisi setiap lokasi pada ruang dua dimensi.

```
[[0.         0.75333749 0.4793706  ... 0.3910632  0.41899608 0.49826235]
 [0.75333749 0.         0.75399884 ... 0.92674199 0.36777901 0.25902832]
 [0.4793706  0.75399884 0.         ... 0.25486708 0.62110708 0.60038297]
 ...
 [0.3910632  0.92674199 0.25486708 ... 0.         0.69904573 0.72134792]
 [0.41899608 0.36777901 0.62110708 ... 0.69904573 0.         0.12084807]
 [0.49826235 0.25902832 0.60038297 ... 0.72134792 0.12084807 0.        ]]
```

Gambar 1. Dataset *Traveling Salesman Problem*

Struktur Dataset:

1. Jumlah Data: 99 titik atau lokasi yang akan menjadi tujuan kunjungan.
2. Kolom Koordinat: Dataset memiliki dua kolom numerik (tanpa nama khusus) yang masing-masing berisi nilai koordinat x dan y untuk setiap lokasi.
3. Tipe Data: Kedua kolom bertipe *float64*, menunjukkan bahwa nilai koordinat memiliki presisi desimal yang diperlukan untuk perhitungan jarak yang akurat.

Setiap baris dalam dataset ini memberikan posisi satu titik dalam ruang dua dimensi. Algoritma akan menggunakan koordinat-koordinat ini untuk menghitung jarak antar titik menggunakan rumus jarak Euclidean. Jarak *Euclidean* antar titik ini kemudian digunakan untuk menentukan rute terpendek bagi salesman. Dalam implementasi Algoritma Genetika, dataset ini berfungsi sebagai input untuk menentukan jarak tempuh dalam setiap solusi rute, yang selanjutnya dievaluasi untuk mendapatkan solusi optimal.

3. HASIL DAN PEMBAHASAN

3.1. Parameter Algoritma

Pada bagian ini disajikan hasil eksperimen Algoritma Genetika (AG) yang digunakan untuk menyelesaikan masalah *Traveling Salesman Problem* (TSP). Hasil ini mencakup parameter yang digunakan dalam algoritma, matriks jarak yang diperoleh dari dataset, implementasi algoritma, serta analisis hasil rute terbaik yang ditemukan.

1. Ukuran Populasi: Kami menetapkan ukuran populasi sebanyak 50 individu. Dengan populasi berukuran sedang ini, kami berusaha untuk mencapai keseimbangan antara variasi solusi dalam populasi dan efisiensi komputasi. Ukuran populasi yang terlalu kecil dapat mengakibatkan kurangnya variasi, yang pada akhirnya membatasi algoritma dalam menemukan solusi optimal. Sebaliknya, ukuran populasi yang terlalu besar dapat meningkatkan beban komputasi tanpa jaminan peningkatan performa yang signifikan.
2. Jumlah Generasi: Kami menjalankan AG selama 100 generasi. Parameter ini ditetapkan dengan harapan bahwa 100 generasi cukup untuk memungkinkan algoritma mendekati atau mencapai solusi optimal. Pada setiap generasi, individu dalam populasi berevolusi melalui proses seleksi, *crossover*, dan mutasi, yang bertujuan untuk memperbaiki kualitas solusi secara bertahap. Jumlah generasi yang terlalu sedikit dapat menyebabkan algoritma belum mencapai solusi optimal, sementara jumlah generasi yang terlalu banyak bisa memperpanjang waktu komputasi tanpa memberikan peningkatan signifikan dalam kualitas solusi.
3. Tingkat Mutasi: Tingkat mutasi ditetapkan pada 10%. Mutasi bertujuan untuk memperkenalkan variasi genetik dalam populasi, sehingga mencegah algoritma terkunci pada solusi optimal lokal. Dengan tingkat mutasi sebesar 10%, kami berharap dapat mempertahankan keragaman dalam populasi tanpa mengganggu proses konvergensi algoritma menuju solusi optimal. Mutasi yang terlalu rendah berisiko menyebabkan solusi optimal terlewat, sedangkan mutasi yang terlalu tinggi bisa mengurangi stabilitas populasi.
4. Tingkat *Crossover*: Tingkat *crossover* ditetapkan sebesar 80%. Parameter ini berarti bahwa pada setiap generasi, 80% individu dalam populasi akan mengalami proses *crossover* untuk menghasilkan keturunan baru. *Crossover* memungkinkan pertukaran informasi genetik antara dua individu (solusi) terpilih, sehingga kombinasi sifat yang lebih baik dari setiap solusi dapat diwariskan ke generasi berikutnya. Dengan tingkat *crossover* yang cukup tinggi, kami berharap mayoritas individu dalam generasi baru akan memiliki kombinasi sifat yang mendekati optimal.

3.2. Matriks Jarak

Dalam proses evaluasi solusi, kami menggunakan matriks jarak yang dihitung berdasarkan jarak *Euclidean* antar titik dalam dataset. Matriks jarak ini memainkan peran penting dalam perhitungan *fitness* setiap solusi dalam populasi, karena *fitness* ditentukan oleh total jarak yang harus ditempuh dalam rute yang dihasilkan oleh setiap Solusi [8]. Matriks jarak memungkinkan Algoritma Genetika menghitung jarak total dengan cepat dan efisien untuk setiap solusi tanpa perlu menghitung ulang jarak antar titik setiap kali evaluasi dilakukan.

Jarak antar titik dihitung menggunakan rumus *Euclidean* sebagai berikut [9]:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (1)$$

di mana x_1, y_1 dan x_2, y_2 adalah koordinat dua titik yang berbeda. Hasil dari perhitungan ini disusun dalam bentuk matriks yang merepresentasikan jarak langsung antar semua pasangan titik. Matriks jarak ini mempermudah Algoritma Genetika dalam melakukan evaluasi *fitness* setiap solusi dengan cepat pada setiap generasi, sehingga mempercepat proses seleksi dan evolusi populasi [10].

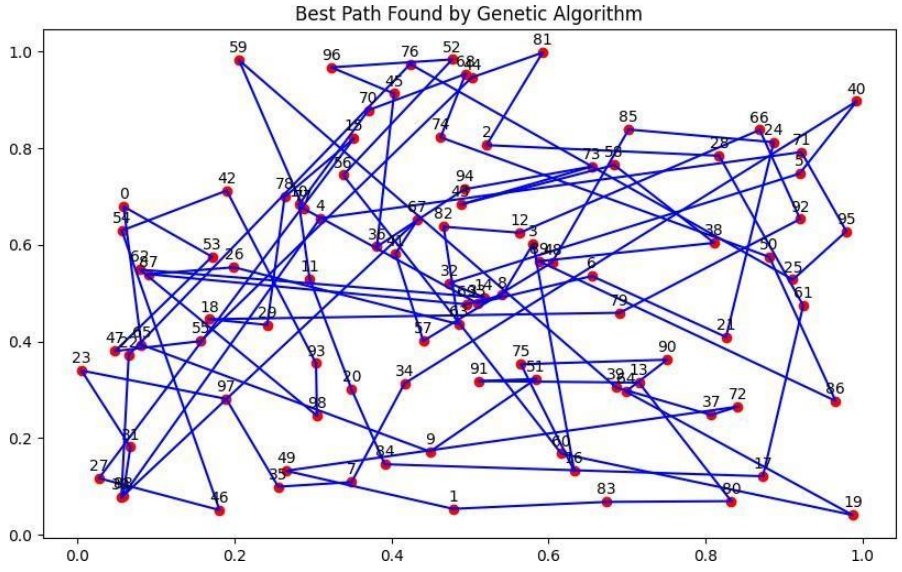
3.3 Implementasi Algoritma

Algoritma diimplementasikan menggunakan Python, dengan pustaka SciPy untuk perhitungan jarak *Euclidean* dan Matplotlib untuk visualisasi rute terbaik yang ditemukan. Dataset koordinat lokasi dimasukkan sebagai input untuk AG, yang kemudian mengolahnya untuk menghasilkan rute yang mendekati optimal. Berikut tahapan utama implementasi:

1. Inisialisasi Populasi: Algoritma memulai dengan membentuk populasi awal berisi 50 solusi acak, masing-masing merupakan urutan kunjungan kota yang unik.
2. Seleksi (Tournament Selection): Algoritma menggunakan metode tournament selection di mana tiga individu dari populasi dipilih secara acak untuk berkompetisi, dan individu dengan nilai *fitness* terbaik dipilih untuk reproduksi.
3. *Crossover* dan Mutasi: Pada tahap *crossover*, AG menggunakan metode *Partially Mapped Crossover* (PMX), yang menjaga sebagian urutan kota dari kedua orang tua dalam keturunan. Mutasi dilakukan dengan metode swap mutation, di mana dua titik dalam solusi dipertukarkan untuk menjaga keragaman dalam populasi.

3.4 Hasil Eksperimen

Setelah menjalankan Algoritma Genetika selama 100 generasi, algoritma menghasilkan rute terbaik dengan total jarak terpendek sebesar 28.19. Berikut adalah urutan solusi terbaik yang ditemukan:
[61, 28, 2, 81, 44, 55, 47, 53, 0, 65, 9, 51, 91, 13, 80, 83, 1, 49, 72, 37, 39, 59, 77, 33, 6, 21, 24, 85, 48, 57, 41, 67, 30, 22, 76, 25, 95, 71, 4, 88, 31, 23, 97, 35, 7, 34, 40, 5, 32, 14, 62, 98, 93, 42, 54, 46, 27, 15, 78, 29, 18, 79, 92, 66, 12, 82, 63, 26, 87, 69, 8, 3, 16, 75, 90, 64, 19, 60, 56, 52, 96, 45, 36, 94, 73, 43, 58, 38, 89, 86, 50, 74, 68, 70, 10, 11, 20, 84, 17].



Gambar 2. Urutan Solusi Terbaik

Total jarak tempuh sebesar 28.19 ini menunjukkan efektivitas algoritma dalam menemukan solusi optimal atau mendekati optimal. Rute ini meminimalkan jarak tempuh yang harus ditempuh untuk mengunjungi setiap titik hanya sekali sebelum kembali ke titik awal.

3.5 Pembahasan

Hasil eksperimen menunjukkan bahwa Algoritma Genetika efektif dalam menyelesaikan masalah TSP. Dengan pengaturan parameter yang tepat, AG berhasil mencapai solusi yang mendekati optimal dalam batasan jumlah generasi yang telah ditentukan. Matriks jarak *Euclidean* yang dihitung di awal memberikan efisiensi komputasi dalam evaluasi *fitness* setiap solusi, sehingga proses optimasi berjalan dengan cepat dan efektif.

Visualisasi rute terbaik menunjukkan bahwa Algoritma Genetika mampu menghasilkan jalur kunjungan kota yang logis dan efisien, dengan minimal persilangan yang menunjukkan jalur terpendek antara titik-titik. Performa algoritma dapat dipengaruhi oleh pemilihan parameter seperti ukuran populasi, jumlah generasi, tingkat *crossover*, dan tingkat mutasi. Eksperimen lebih lanjut dengan variasi parameter ini dapat mengungkap pengaruh parameter tersebut terhadap kualitas hasil, khususnya pada kasus dengan jumlah titik lebih besar atau konfigurasi rute yang lebih kompleks. Secara keseluruhan, hasil eksperimen ini menunjukkan bahwa Algoritma Genetika adalah metode yang efisien untuk menyelesaikan TSP, menghasilkan solusi yang optimal atau mendekati optimal dengan waktu komputasi yang dapat diterima.

4. KESIMPULAN

Penelitian ini menunjukkan bahwa Algoritma Genetika (AG) merupakan metode yang efektif untuk menyelesaikan masalah *Traveling Salesman Problem* (TSP), di mana AG mampu menemukan solusi rute yang mendekati optimal dengan waktu komputasi yang efisien. Dengan memanfaatkan proses seleksi, *crossover*, dan mutasi, AG memungkinkan pencarian solusi yang terus berkembang dan mendekati optimal di setiap generasi. Pengaturan parameter yang tepat, termasuk ukuran populasi, jumlah generasi, tingkat *crossover*, dan tingkat mutasi, berkontribusi signifikan dalam mengoptimalkan performa algoritma. Parameter-parameter tersebut menjaga keseimbangan antara eksplorasi dan eksploitasi solusi, serta mencegah algoritma terkunci pada solusi optimal lokal. Penggunaan matriks jarak *Euclidean* dalam perhitungan jarak antar titik terbukti efektif dalam mempercepat proses evaluasi *fitness*, sehingga algoritma dapat memusatkan sumber daya pada proses optimasi populasi solusi.

Eksperimen ini mengonfirmasi bahwa AG dapat menghasilkan rute yang efisien untuk mengunjungi seluruh titik pada dataset dengan jarak tempuh yang minimum. Visualisasi rute terbaik yang diperoleh menunjukkan bahwa rute yang dihasilkan tidak hanya optimal secara matematis, tetapi juga logis secara visual dengan minimnya persilangan jalur. Hasil ini menunjukkan potensi AG sebagai pendekatan yang aplikatif dan efektif untuk optimasi rute, khususnya dalam skenario nyata seperti perencanaan distribusi dan logistik.

REFERENSI

[1] R. M. F. Alves and C. R. Lopes, "Using genetic algorithms to minimize the distance and balance the routes for the multiple *Traveling Salesman Problem*," in *2015 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, May 2015, pp. 3171–3178. doi: 10.1109/CEC.2015.7257285.

[2] B. A. Ajayi, M. A. Magaji, S. Musa, R. F. Olanrewaju, and A. A. Salihu, "A Comparative Analysis of Optimization Heuristics Algorithms as Optimal Solution for Travelling Salesman Problem," in *2022 5th Information Technology for Education and Development (ITED)*, IEEE, Nov. 2022, pp. 1–8. doi: 10.1109/ITED56637.2022.10051627.

[3] S. Kavita and S. S. K., "Metaheuristic Evolutionary Algorithms: Types, Applications, Future Directions, and Challenges," in *2023 3rd International Conference on Intelligent Technologies (CONIT)*, IEEE, Jun. 2023, pp. 1–6. doi: 10.1109/CONIT59222.2023.10205592.

[4] S. Prayudani, A. Hizriadi, E. B. Nababan, and S. Suwilo, "Analysis Effect of Tournament Selection on Genetic Algorithm Performance in *Traveling Salesman Problem* (TSP)," *J Phys Conf Ser*, vol. 1566, no. 1, p. 012131, Jun. 2020, doi: 10.1088/1742-6596/1566/1/012131.

[5] A. M. Aibinu, H. Bello Salau, N. A. Rahman, M. N. Nwohu, and C. M. Akachukwu, "A novel Clustering based Genetic Algorithm for route optimization," *Engineering Science and Technology, an International Journal*, vol. 19, no. 4, pp. 2022–2034, Dec. 2016, doi: 10.1016/j.jestch.2016.08.003.

[6] J. Wu and S. Feng, "Improved biogeography-based optimization for the *traveling salesman problem*," in *2017 2nd IEEE International Conference on Computational Intelligence and Applications (ICCIA)*, IEEE, Sep. 2017, pp. 166–171. doi: 10.1109/CIAPP.2017.8167201.

[7] Y.-F. CHEN and X.-H. WU, "Improved Genetic Algorithm for Optimizing TSP Problem," *DEStech Transactions on Social Science, Education and Human Science*, no. iced, Nov. 2017, doi: 10.12783/dtssehs/iced2017/15154.

[8] S. Dhouib, "A New Column-Row Method for *Traveling Salesman Problem*: The Dhouib-Matrix-TSP1," *International Journal of Recent Engineering Science*, vol. 8, no. 1, pp. 6–10, Feb. 2021, doi: 10.14445/23497157/IJRES-V8I1P102.

[9] A. Bertagnon and M. Gavanelli, "Improved Filtering for the *Euclidean* Traveling Salesperson Problem in CLP(FD)," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 02, pp. 1412–1419, Apr. 2020, doi: 10.1609/aaai.v34i02.5498.

[10] H. Du, Z. Wang, W. Zhan, and J. Guo, "Elitism and Distance Strategy for Selection of Evolutionary Algorithms," *IEEE Access*, vol. 6, pp. 44531–44541, 2018, doi: 10.1109/ACCESS.2018.2861760.