

# Perbandingan Algoritma *Simulated Annealing*, *Genetic Algorithm*, dan *Ant Colony Optimization* terhadap Permasalahan *Travelling Salesman Problem*

Faris Eka Darmawan<sup>1</sup> Safuan<sup>2</sup>

<sup>1</sup>Program Studi Informatika, Fakultas Teknik Universitas Muhammadiyah Semarang, Semarang, Indonesia

<sup>2</sup>Program Studi Informatika, Fakultas Teknik Universitas Muhammadiyah Semarang, Semarang, Indonesia

[farisdermawan63@gmail.com](mailto:farisdermawan63@gmail.com)

[safuan@unimus.ac.id](mailto:safuan@unimus.ac.id)

## Info Artikel

### Riwayat Artikel:

Diterima 15, Oktober, 2025  
Perbaikan 22, Desember, 2025  
Disetujui 13, Januari, 2026

### Keywords:

*Travelling Salesman Problem*  
*Genetic Algorithm*  
*Simulated Annealing*  
*Ant Colony Optimization*  
Dataset Berlin

## ABSTRAK

Percobaan ini membandingkan tiga algoritma heuristik, yaitu *Genetic Algorithm* (GA), *Simulated Annealing* (SA), dan *Ant Colony Optimization* (ACO), dalam menyelesaikan masalah *Travelling Salesman Problem* (TSP) menggunakan dataset Berlin. TSP merupakan masalah kombinatorial yang bertujuan untuk menentukan rute terpendek yang mengunjungi sejumlah titik dan kembali ke titik awal. Metode percobaan mencakup penerapan ketiga algoritma dengan pengaturan parameter yang konsisten, termasuk jumlah iterasi sebanyak 10 kali, untuk mengevaluasi kinerja masing-masing algoritma. Hasil percobaan menunjukkan bahwa GA mencapai solusi terbaik dengan total jarak tempuh 27.539,86, diikuti oleh ACO dengan jarak 28.774,06, dan SA dengan jarak 28.897,26. Analisis hasil menunjukkan bahwa GA secara efektif mengeksplorasi ruang solusi, menghasilkan rute yang lebih efisien dibandingkan kedua algoritma lainnya. Selain itu, percobaan ini memberikan rekomendasi untuk pengembangan lebih lanjut, termasuk tuning parameter dan penerapan algoritma pada skala yang lebih besar. Temuan ini diharapkan dapat memberikan wawasan yang berharga untuk implementasi algoritma heuristik dalam memecahkan masalah optimasi dalam dunia nyata.

## ABSTRACT

This experiment compares three heuristic algorithms, namely *Genetic Algorithm* (GA), *Simulated Annealing* (SA), and *Ant Colony Optimization* (ACO), in solving the *Travelling Salesman Problem* (TSP) using the Berlin dataset. The *Travelling Salesman Problem* (TSP) is a combinatorial problem that aims to determine the shortest route that visits a number of points and returns to the starting point. The experimental method entails the application of the three algorithms with consistent parameter settings, including the number of iterations, 10 times, to evaluate the performance of each algorithm. The experimental results demonstrated that the GA achieved the optimal solution with a total distance traveled of 27,539.86, followed by the ACO with a distance of 28,774.06, and the SA with a distance of 28,897.26. The analysis of the results demonstrates that GA effectively explores the solution space, thereby producing a more efficient route than the other two algorithms. Furthermore, this experiment offers recommendations for future development, including parameter tuning and the application of the algorithm on a larger scale. It is anticipated that the findings will provide valuable insights for the implementation of heuristic algorithms in solving real-world optimization problems.

Ini adalah artikel akses terbuka di bawah lisensi CC BY-SA.



---

**Penulis Korespondensi:**

Faris Eka Darmawan

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 travelling salesman (TSP) adalah salah satu masalah optimasi kombinatorial paling terkenal yang telah dipelajari secara luas selama beberapa dekade. TSP dapat digambarkan sebagai masalah bagi penjual yang ingin mengunjungi sekumpulan kota, mengunjungi setiap kota satu kali dan menemukan rute optimal untuk kembali ke kota asal dengan jarak perjalanan minimum. Meskipun konsepnya terlihat sederhana, TSP merupakan permasalahan NP-hard. Hal ini berarti bahwa di banyak kota, jumlah rute yang memungkinkan meningkat secara eksponensial, sehingga sangat sulit untuk menemukan solusi optimal dengan cara yang efisien waktu. TSP mempunyai banyak kegunaan praktis dalam kehidupan sehari-hari, seperti : Rute pengiriman produk (logistik), Rute perjalanan wisata, Pengiriman produk, dan Jalur robot manufaktur.

Permasalahan Berlin TSP adalah salah satu contoh terkenal dari dataset TSP, yang terdiri dari 52 kota yang terletak di Berlin, Jerman. Setiap kota diwakili oleh koordinat kartesian (x, y), dan tujuan dari permasalahan ini adalah mencari rute optimal untuk mengunjungi semua kota ini dan kembali ke titik awal. Dataset Berlin TSP sering digunakan sebagai standar benchmark dalam pengujian algoritma optimasi karena skalanya yang menengah, serta memiliki karakteristik spasial yang menantang.

Dalam menyelesaikan TSP, terutama ketika jumlah kota atau titik sangat besar, solusi optimal secara eksak sulit ditemukan. Oleh karena itu, algoritma heuristik digunakan untuk mencari solusi yang mendekati optimal dalam waktu yang lebih efisien. Tiga algoritma yang digunakan dalam percobaan ini adalah: *Simulated Annealing* (SA), *Genetic Algorithm* (GA), dan *Ant Colony Optimization* (ACO).

Algoritma *Simulated Annealing* (SA) terinspirasi dari proses fisik peleburan logam, di mana sistem dibiarkan untuk mendingin perlahan agar mencapai kondisi energi minimum. Dalam konteks TSP, SA bekerja dengan mengacak solusi awal, kemudian bertransisi secara bertahap ke solusi yang lebih baik, tetapi tetap mempertimbangkan kemungkinan menerima solusi yang sedikit lebih buruk untuk keluar dari local minima. Semakin lama proses berjalan, kemungkinan untuk menerima solusi buruk menurun, hingga akhirnya konvergen pada solusi yang mendekati optimal.

*Genetic Algorithm* (GA) adalah metode optimasi yang terinspirasi oleh proses evolusi alam. Dalam algoritma ini, solusi dianggap sebagai "individu" yang memiliki "gen" (representasi dari solusi). Algoritma ini bekerja melalui seleksi alam dengan mengkombinasikan, melakukan mutasi, dan merekombinasi solusi untuk menghasilkan solusi yang lebih baik dari generasi ke generasi. Dalam TSP, solusi yang mendekati rute terpendek akan dipertahankan dan dikombinasikan untuk menghasilkan solusi yang lebih optimal.

*Ant Colony Optimization* (ACO) terinspirasi oleh perilaku koloni semut dalam mencari jalur terpendek antara sarang dan sumber makanan. Setiap semut secara acak memilih rute dengan probabilitas yang dipandu oleh jejak feromon yang ditinggalkan oleh semut lainnya. Semut-semut yang menemukan rute terpendek akan meninggalkan jejak feromon lebih kuat, yang pada akhirnya memandu semut lainnya untuk memilih rute tersebut. Pada TSP, algoritma ini dapat digunakan untuk menemukan rute optimal yang mendekati solusi terbaik. Ketiga algoritma yang dipilih (SA, GA, dan ACO) memiliki karakteristik yang berbeda dalam proses pencarian solusi. Masing-masing algoritma memiliki kelebihan dan kekurangan yang membuatnya menarik untuk dibandingkan dalam menyelesaikan TSP, khususnya dataset Berlin52.

## 2. METODE

Metode yang digunakan pada percobaan yang akan dilakukan mencakup tiga algoritma yaitu *Simulated Annealing* (SA), *Genetic Algorithm* (GA), *Ant Colony Optimization* (ACO).

### 2.1. *Simulated Annealing* (SA)

*Simulated Annealing* (SA) meniru proses *annealing* dalam metalurgi, di mana material dipanaskan dan kemudian didinginkan secara perlahan untuk mencapai keadaan stabil dengan energi minimal. Dalam TSP,

algoritma ini secara iteratif menukar kota dalam rute dan memutuskan apakah akan menerima rute baru berdasarkan probabilitas yang menurun seiring penurunan suhu. :

- 1) Inisialisasi  
Memulai dengan solusi awal dan suhu tinggi
- 2) Pemilihan Tetangga  
Menukar dua kota secara acak untuk menghasilkan rute baru.
- 3) Kriteria Penerimaan  
Jika rute baru lebih pendek, terima; jika tidak, terima dengan probabilitas yang tergantung pada suhu.
- 4) Jadwal Pendinginan  
Menurunkan suhu berdasarkan tingkat pendinginan
- 5) Terminasi  
Berhenti ketika suhu di bawah ambang batas tertentu.

## 2.2. Genetic Algorithm (GA)

*Genetic Algorithm* (GA) terinspirasi oleh proses seleksi alam. Algoritma ini dimulai dengan populasi solusi potensial dan secara iteratif menerapkan operasi *crossover* dan mutasi untuk menciptakan generasi baru, dengan tujuan meningkatkan fitness dari populasi tersebut. Dalam permasalahan TSP, setiap individu dalam populasi direpresentasikan sebagai permutasi kota. Jarak total perjalanan berfungsi sebagai fungsi *fitness* yang harus diminimalkan.

Langkah-langkah GA:

- 1) Inisialisasi  
Membuat populasi awal rute acak.
- 2) Seleksi  
Memilih dua rute orang tua menggunakan seleksi turnamen.
- 3) *Crossover*  
Menghasilkan keturunan dengan *crossover* yang dipetakan sebagian (PMX).
- 4) Mutasi  
Menukar kota dalam rute dengan probabilitas mutasi tertentu.
- 5) Evaluasi  
Menghitung jarak total setiap rute.
- 6) Penggantian  
Menggantikan populasi dengan generasi baru.

## 2.3. Ant Colony Optimization (ACO)

*Ant Colony Optimization* (ACO) terinspirasi oleh perilaku semut dalam mencari makanan. Semut meninggalkan feromon di sepanjang jalur yang mereka lalui, yang memengaruhi semut lain untuk mengikuti jalur tersebut. Dalam TSP, semut buatan membangun solusi dengan memilih kota secara probabilistik berdasarkan tingkat feromon dan informasi heuristik (kebalikan dari jarak).

Langkah-langkah ACO:

- 1) Inisialisasi  
Menetapkan tingkat feromon awal antara kota.
- 2) Konstruksi Solusi  
Semut membangun rute menggunakan aturan probabilistik berdasarkan feromon dan informasi heuristik.
- 3) Pembaruan Feromon  
Meningkatkan tingkat feromon pada jalur yang digunakan oleh rute terpendek dan menguapkan feromon pada jalur lain.
- 4) Konvergensi  
Mengulangi hingga tingkat feromon stabil atau jumlah iterasi maksimum tercapai.

## 3. DATA DAN IMPLEMENTASI

### 3.1. Data

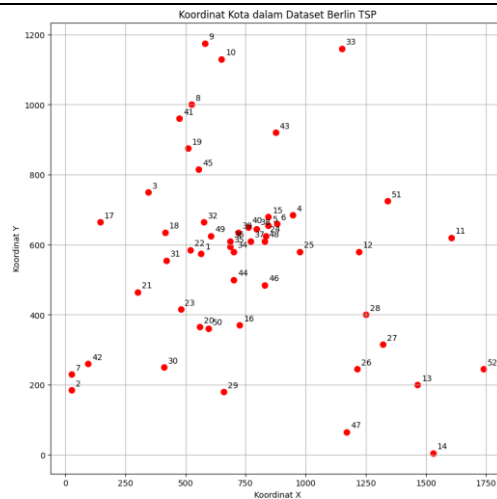
Data yang digunakan dalam proses percobaan ini merupakan data nyata yaitu Dataset Berlin TSP. Berlin TSP adalah dataset *benchmark* yang terkenal untuk masalah *Travelling Salesman Problem*. Dataset ini

berisi informasi tentang lokasi-lokasi tertentu (dalam hal ini, kota-kota) yang harus dikunjungi yang terdiri dari 52 lokasi yang mewakili titik-titik koordinat di Berlin, Jerman. Dataset ini pertama kali dibuat oleh Groetschel. Dataset ini beresumber dari <http://comopt.ifl.uniheidelberg.de/software/TSPLIB95/> atau melalui <https://github.com/pdroszdowski/TSPLib.Net/blob/master/TSPLIB95/tsp/berlin52.tsp>.

Dataset Berlin TSP memiliki rincian sebagai berikut:

1) Jumlah Titik (Kota)

Dataset Berlin TSP terdiri dari 52 titik yang mewakili 52 kota. Masing-masing kota memiliki koordinat geografis (x, y) yang menunjukkan posisinya di peta seperti pada *Gambar 1*.



*Gambar*

*1. Titik Koordinat Kota dalam Dataset Berlin*

2) Format Data

Setiap titik dalam dataset ditampilkan dalam format: [ID, *x-coordinate*, *y-coordinate*]

a) ID

Nomor identifikasi kota (1 hingga 52).

b) *x-coordinate*

Koordinat horizontal kota dalam satuan tertentu (*kilometer*).

c) *y-coordinate*

Koordinat vertikal kota dalam satuan yang sama.

3) Contoh Data

Berikut merupakan contoh data dari dataset Berlin TSP:

[1, 565.0, 575.0]

[2, 25.0, 185.0]

[3, 345.0, 750.0]

[4, 945.0, 685.0]

...

[52, 1740.0, 245.0]

4) Jarak Antara Lokasi

Jarak antara dua kota dihitung menggunakan rumus *Euclidean* seperti pada (1):

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

Dengan menggunakan koordinat (x, y) masing-masing kota, jarak antar lokasi dapat dihitung untuk menentukan rute terpendek yang harus dilalui

5) Penggunaan dalam TSP

Tujuan dari TSP adalah untuk menemukan rute terpendek yang mengunjungi semua kota tepat satu kali dan kembali ke kota asal. Dalam hal ini, dataset Berlin TSP menjadi kasus uji yang baik untuk

menguji algoritma seperti *Genetic Algorithm* (GA), *Simulated Annealing* (SA), dan *Ant Colony Optimization* (ACO).

### 3.2. Implementasi

- 1) Jumlah Iterasi
  - a) Jumlah iterasi yang digunakan dalam setiap algoritma adalah 10 iterasi.
  - b) Setiap iterasi dirancang untuk mengoptimalkan rute kendaraan dan mengurangi jarak tempuh total. Hasil dari setiap iterasi akan dianalisis untuk mendapatkan rute terbaik.
- 2) Jumlah Semut (ACO)
  - a) Dalam algoritma *Ant Colony Optimization*, jumlah semut yang digunakan dalam simulasi adalah 10 semut. Setiap semut akan mencoba menemukan rute dari titik awal ke titik akhir dan mengunjungi semua titik (kota) dalam dataset.
  - b) Setiap semut bekerja secara independen untuk menemukan solusi, dan hasil yang didapatkan dari setiap semut akan digunakan untuk memperbarui informasi feromon di antara jalur yang telah dilalui. Dengan cara ini, jalur yang lebih pendek akan mendapatkan lebih banyak feromon, sehingga lebih mungkin dipilih oleh semut-semut di iterasi selanjutnya .
- 3) Tingkat Evaporasi Feromon

Parameter ini menentukan seberapa cepat feromon menguap dari jalur. Nilai ini biasanya berkisar antara 0 dan 1, dan semakin rendah nilainya, semakin cepat feromon menguap. Dalam simulasi ini, digunakan nilai 0.5 .

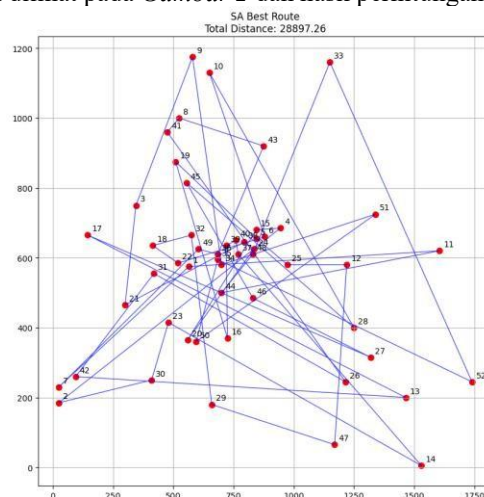
## 4. HASIL DAN PEMBAHASAN

Setelah dilakukan proses simulasi dengan menggunakan bahasa pemrograman *python* pada aplikasi *Google Colaboratory*, maka diperoleh hasil dengan rincian sebagai berikut:

### 4.1. Rute yang Ditemukan oleh Setiap Algoritma

- 1) *Simulated Annealing* (SA)

Hasil simulasi visual dapat dilihat pada *Gambar 2* dan hasil perhitungan pada *Tabel 1*.



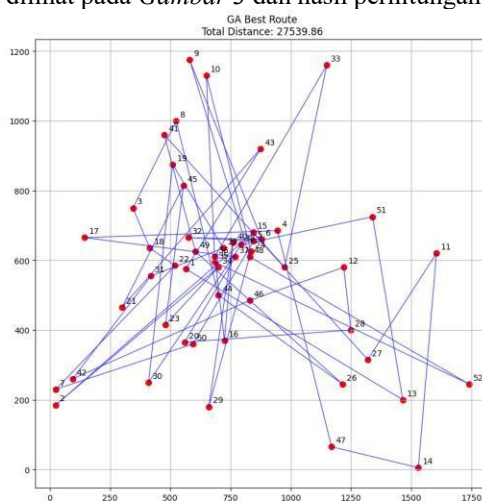
Gambar 2. Visualisasi Simulasi Rute dengan *Simulated Annealing*

Tabel 1. Hasil Perhitungan dengan *Simulated Annealing*

Best Distance	28897.26
Best Route	[30, 25, 40, 7, 42, 35, 48, 6, 0, 10, 43, 47, 32, 51, 34, 49, 50, 33, 8, 2, 20, 37, 23, 19, 14, 3, 1, 29, 22, 13, 36, 39, 38, 15, 18, 4, 24, 11, 46, 28, 31, 17, 21, 5, 9, 27, 44, 45, 26, 16, 12, 41]
Average Distance	30530.98 $\pm$ 814.27
Average Time	0.0007 seconds $\pm$ 0.0004 second

## 2) Genetic Algorithm (GA)

Hasil simulasi visual dapat dilihat pada *Gambar 3* dan hasil perhitungan pada *Tabel 2*.



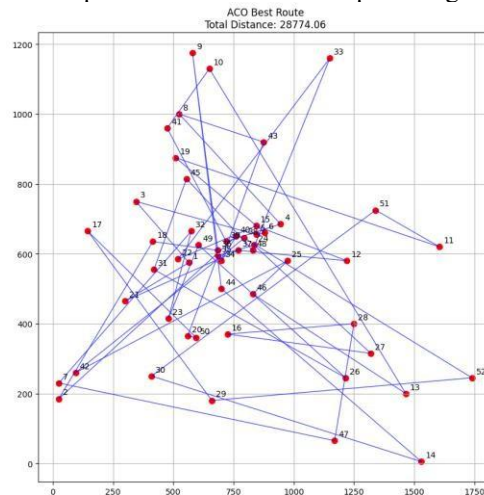
*Gambar 3. Visualisasi Simulasi Rute dengan Genetic Algorithm*

*Tabel 2. Hasil Perhitungan dengan Genetic Algorithm*

Best Distance	27539.86
Best Route	[26, 40, 48, 25, 43, 7, 2, 17, 21, 6, 49, 4, 31, 5, 8, 47, 51, 34, 9, 23, 28, 39, 15, 35, 32, 24, 14, 1, 33, 22, 44, 20, 42, 0, 12, 50, 37, 29, 18, 38, 30, 41, 45, 11, 27, 19, 36, 16, 3, 46, 13, 10]
Average Distance	30667.85 $\pm$ 1661.04
Average Time	0.0006 seconds $\pm$ 0.0001 seconds

3) *Ant Colony Optimization (ACO)*

Hasil simulasi visual dapat dilihat pada *Gambar 4* dan hasil perhitungan pada *Tabel 3*.



*Gambar 4. Visualisasi Simulasi Rute dengan Ant Colony Optimization*

*Tabel 3. Hasil Perhitungan dengan Ant Colony Optimization*

Best Distance	28774.06
Best Route	[50, 10, 18, 5, 4, 2, 0, 20, 36, 11, 14, 21, 32, 47, 44, 30, 25, 48, 22, 31, 6, 46, 27, 15, 26, 23, 34, 19, 49, 16, 28, 51, 37, 41, 24, 45, 12, 9, 40, 38, 3, 7, 42, 33, 39, 1, 17, 35, 8, 43, 13, 29]
Average Distance	30543.00 ± 1168.13
Average Time	0.0006 seconds ± 0.0001 seconds

#### 4.2. Perbandingan Jarak Tempuh dan Rute

Hasil yang diperoleh menunjukkan bahwa *Genetic Algorithm* berhasil menemukan rute dengan jarak terpendek (27539.86) dibandingkan dengan *Simulated Annealing* (28897.26) dan *Ant Colony Optimization* (28774.06).

#### 4.3. Efisiensi Jarak Tempuh

Semua algoritma menunjukkan kinerja yang baik dalam menghasilkan solusi. Namun, jarak rata-rata yang dihasilkan oleh semua metode relatif dekat, dengan GA yang mencatatkan jarak rata-rata tertinggi tetapi dengan fluktuasi yang lebih besar ( $\pm 1661.04$ ), sedangkan SA memiliki fluktuasi lebih rendah ( $\pm 814.27$ ). Ini menunjukkan bahwa SA mungkin lebih stabil dalam menemukan solusi dalam rentang iterasi yang diberikan.

#### 4.4. Analisis Kecepatan Konvergensi

- 1) *Genetic Algorithm*(GA) menunjukkan kecepatan konvergensi yang baik dalam menemukan solusi optimal, tetapi dapat memerlukan lebih banyak iterasi untuk mencapai hasil yang stabil.
- 2) *Simulated Annealing*(SA) cenderung lebih lambat dalam konvergensi ke solusi optimal, tetapi memberikan stabilitas yang lebih baik dalam hasil jarak rata-rata.
- 3) *Ant Colony Optimization*(ACO) juga menunjukkan kecepatan konvergensi yang baik, dengan hasil yang bersaing, namun tidak sebaik GA.

#### 4.5. Keterbatasan

*Genetic Algorithm* (GA) bisa saja terjebak di solusi lokal jika variasi dalam populasinya kurang. *Simulated Annealing* (SA) sangat bergantung pada pengaturan suhu, yang perlu di-*tuning* dengan cermat supaya hasilnya bagus. Sementara itu, *Ant Colony Optimization* (ACO) butuh pengaturan parameter yang tepat, seperti tingkat penguapan *feromon*, supaya performanya optimal.

### 5. KESIMPULAN

Berdasarkan perbandingan tiga algoritma yaitu *Genetic Algorithm*(GA), *Simulated Annealing*(SA), dan *Ant Colony Optimization*(ACO) dalam menyelesaikan Travelling Salesman Problem (TSP) menggunakan dataset Berlin, GA menunjukkan kinerja paling unggul dengan rute terbaik mencapai 27.539,86, diikuti oleh ACO dengan jarak 28.774,06, dan SA dengan 28.897,26.

Keunggulan GA dapat dikaitkan dengan karakteristiknya yang efektif dalam mengeksplorasi ruang solusi besar melalui mekanisme seleksi, perkawinan, dan mutasi, memungkinkannya mengeksplorasi berbagai kemungkinan rute secara simultan. Sementara itu, ACO menunjukkan performa yang bersaing berkat kemampuannya mensimulasikan perilaku semut dalam mencari rute optimal menggunakan *feromon*, dan SA, meskipun lebih sederhana, tetap mampu memberikan solusi yang memadai melalui pendekatan probabilistik dalam menjelajahi ruang solusi, menjadikan ketiga algoritma ini pilihan yang baik untuk menyelesaikan TSP, dengan GA sebagai pilihan utama berdasarkan hasil yang diperoleh.

### REFERENSI

- [1] G. Gutin and A. P. Punnen, *The Traveling Salesman Problem and Its Variations*. New York, NY, USA: Springer Science & Business Media, 2002.
- [2] E. L. Lawler, J. K. Lenstra, A. H. Rinnooy Kan, and D. B. Shmoys, *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*. Hoboken, NJ, USA: Wiley, 1985.
- [3] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671-680, May 1983, doi: 10.1126/science.220.4598.671.
- [4] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1992.
- [5] M. Dorigo and T. Stützle, *Ant Colony Optimization*. Cambridge, MA, USA: MIT Press, 2004.
- [6] M. Dorigo and L. M. Gambardella, "Ant colonies for the traveling salesman problem," *BioSystems*, vol. 43, no. 2, pp. 73-81, Jul. 1997, doi: 10.1016/S0303-2647(97)01708-5.
- [7] G. Reinelt, "TSPLIB—A traveling salesman problem library," *ORSA J. Comput.*, vol. 3, no. 4, pp. 376-384, Nov. 1991, doi: 10.1287/ijoc.3.4.376.
- [8] G. Reinelt, "TSPLIB 95: A library of sample instances for the TSP (Travelling Salesman Problem) and related problems," Univ. Heidelberg, Heidelberg, Germany, Tech. Rep., 1995. [Online]. Available: <http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/>