EVALUASI ALGORITMA *ANT COLONY SYSTEM*: PERBANDINGAN DENGAN METODE EKSAK DAN METAHEURISTIK DALAM PENYELESAIAN *TRAVELLING SALESMAN PROBLEM*

Nurani Candra Widyasari, *Bib Paruhum Silalahi, dan Hidayatul Mayyani

Program Studi Matematika, Sekolah Sains Data, Matematika, dan Informatika,
Institut Pertanian Bogor, Jl. Meranti, Kampus IPB Dramaga Bogor.
raniwidyasari@apps.ipb.ac.id, bibparuhum@gmail.com ***corresponding author**,
mayyani_mat15@apps.ipb.ac.id

Abstrak

Salah satu permasalahan di bidang optimasi yang berfokus pada pencarian rute terpendek adalah *Travelling Salesman Problem* (TSP). TSP dikategorikan sebagai masalah NP-*hard* yang artinya tidak ada algoritma yang dapat menyelesaikannya secara optimal dalam waktu polinomial. Pada penelitian ini, TSP diselesaikan menggunakan metode pendekatan *Ant Colony System* (ACS) dan metode eksak *Branch and Bound* (BnB). Solusi akhir dibandingkan dengan metode pendekatan lainnya, yaitu *Bee Colony Optimization* (BCO), *Ant Colony Optimization* (ACO), *Grey Wolf Optimization* (GWO), dan *Simulated Annealing* (SA). Tiga kasus TSP dengan 20, 40, dan 60 kota diselesaikan menggunakan data hipotetik berupa koordinat Kartesius. Hasil menunjukkan bahwa algoritma ACS memiliki kinerja lebih baik dengan solusi yang paling mendekati optimal dan waktu komputasi yang lebih cepat dibandingkan BCO, ACO, GWO, dan SA. Pada kasus 1 dan 2, ACS mencapai solusi optimum global; pada kasus 3, deviasi solusi ACS hanya sebesar 0,368%.

Kata kunci: *ant colony system*, *branch and bound*, metode metaheuristik, *travelling salesman problem*

1 Pendahuluan

Distribusi merupakan proses pemasaran yang bertujuan untuk memperlancar penyaluran barang dan jasa dari produsen ke konsumen [1]. Efektivitas distribusi sangat dipengaruhi oleh pemilihan rute yang tepat, karena rute yang tidak optimal dapat meningkatkan biaya bahan bakar, waktu pengiriman, serta menurunnya produktivitas perusahaan [2]. Salah satu permasalahan di bidang optimasi yang berfokus pada pencarian rute terpendek dalam proses distribusi adalah *Travelling Salesman Problem* (TSP) [3].

TSP merupakan permasalahan optimasi rute terpendek yang dipelajari dalam ilmu komputer dan riset operasi. Dalam TSP, seorang *salesman* harus mengunjungi sejumlah kota dari kota asal, dengan ketentuan setiap kota hanya boleh dilewati tepat satu kali dan kembali ke kota asal setelah semua kota dikunjungi [4]. TSP dikategorikan sebagai masalah NP-*hard* (*Nondeterministic Polynomial-time hard*), yang artinya tidak ada algoritma yang dapat menyelesaikannya secara optimal dalam waktu komputasi polinomial [5].

2020 Mathematics Subject Classification: 90C27, 90C59, 90C35.
Diajukan:02/06/2026, diterima: 24/06/2026. DOI: https://doi.org/10.29244/milang.22.1.53-62
MILANG Journal of Mathematics and Its Applications, Vol.22, No.1, pp.53-62 ISSN: 2963-5233

Permasalahan TSP dapat diselesaikan dengan metode eksak maupun metode pendekatan (heuristik dan metaheuristik). Metode eksak menjamin solusi optimal namun membutuhkan waktu komputasi yang lama untuk kasus besar. Metode heuristik bersifat *problem-dependent*, sedangkan metode metaheuristik bersifat *problem-independent* dan dapat diterapkan pada berbagai jenis masalah [6].

Pada penelitian ini, TSP diselesaikan menggunakan algoritma *Ant Colony System* (ACS), yaitu pengembangan dari algoritma *Ant Colony Optimization* (ACO) yang terinspirasi dari perilaku koloni semut dalam mencari sumber makanannya menggunakan jejak feromon [7]. Penerapan ACO dalam penyelesaian TSP telah dikaji oleh Silalahi *et al.* [8]. Untuk menguji kinerja ACS, solusinya dibandingkan dengan metode BnB dan metode metaheuristik lainnya: BCO, ACO, GWO, dan SA.

2 *Travelling Salesman Problem* (TSP)

TSP awalnya dikaji oleh matematikawan Irlandia William Rowan Hamilton dan matematikawan Inggris Thomas Penyngton Kirkman. Secara formal, TSP melibatkan seorang *salesman* yang harus mengunjungi *n* kota, di mana setiap kota saling terhubung dan memiliki bobot jarak tertentu [9]. Jumlah rute yang mungkin dihitung dengan rumus $(n-1)!/2$.

Menurut Winston [10], TSP dapat diformulasikan sebagai *Integer Linear Programming* (ILP) berikut:

Parameter:

n : banyaknya kota yang akan dikunjungi,

c_{ij} : jarak kota i ke kota j,

u_i : variabel tambahan saat mengunjungi kota i.

Variabel keputusan:

x_{ij} = 1 jika ada perjalanan dari kota *i* ke kota *j*; 0 selainnya.

Fungsi objektif:

$$\min Z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}.$$

Kendala setiap kota dikunjungi tepat satu kali:

$$\sum_{i=1}^{n} x_{ij} = 1, \forall j = 1,2,3,\dots,n. \qquad \sum_{j=1}^{n} x_{ij} = 1, \forall i = 1,2,3,\dots,n.$$

Kendala eliminasi subtur (Miller-Tucker-Zemlin):

$$u_i - u_j + nx_{ij} \leq n-1, i \neq j, \forall i = 2,3,\dots,n;\ \forall j = 2,3,\dots,n; u_j \geq 0.$$

Kendala biner: $x_{ij} \in \{0,1\}, \forall i = 1,2,3,\dots,n;\ \forall j = 1,2,3,\dots,n.$

3 *Ant Colony System* (ACS)

Algoritma ACS merupakan pengembangan dari ACO yang diperkenalkan oleh Dorigo dan Gambardella [7]. Perbedaan utama ACS dengan ACO terletak pada strategi pemilihan jalur dan mekanisme pembaruan tingkat feromon. ACS memiliki tiga karakteristik utama: aturan transisi status, pembaruan tingkat feromon lokal, dan pembaruan tingkat feromon global.

a. Aturan Transisi Status

Semut yang berada di kota t memilih kota tujuan v berdasarkan aturan *pseudo-random-proportional*:

$$v = \begin{cases} \text{argmax}_{u \epsilon J_k(t)} \left(\left(\tau(t,u)\right) \cdot \left(\eta(t,u)\right)^{\beta} \right), & q \leq q_0 \\ p_k(t,u), & \text{lainnya} \end{cases}$$

dengan rumus $p_k(t,u)$ sebagai berikut.

$$p_k(t,u) = \frac{\left(\tau(t,u)\right) \cdot \left(\eta(t,u)\right)^{\beta}}{\sum_{s \epsilon J_k(t)} \left(\tau(t,s)\right) \cdot \left(\eta(t,s)\right)^{\beta}}$$

Keterangan:

v : kota selanjutnya yang terpilih untuk dikunjungi,
$\tau(t,u)$: intensitas feromon dari kota t ke kota u,
$\eta(t,u)$: nilai heuristik yang merupakan invers jarak $d(t,u)$ yaitu jarak dari kota t ke kota u $\left(\eta(t,u) = \frac{1}{d(t,u)}\right)$,
β : parameter yang menentukan kepentingan relatif dari informasi heuristik, yaitu besarnya bobot yang diberikan terhadap parameter tersebut dalam proses pemilihan jalur $(\beta \geq 0)$,
$p_k(t,u)$: probabilitas semut k memilih untuk bergerak dari kota t ke kota u,
$\tau(t,s)$: intensitas feromon dari kota t ke kota s,
$\eta(t,s)$: nilai heuristik yang merupakan invers jarak $d(t,s)$ yaitu jarak dari kota t ke kota s $\left(\eta(t,s) = \frac{1}{d(t,s)}\right)$,
$J_k(t)$: himpunan kota yang belum dikunjungi oleh semut k dari kota t,
q_0 : parameter penentu eksploitasi dan eksplorasi $(0 \leq q_0 \leq 1)$,
q : bilangan acak yang nilainya antara 0 dan 1.

b. Pembaruan Tingkat Feromon Lokal

Setiap kali semut melewati jalur (t,u), nilai feromon diperbarui:

$$\tau(t,u)_{k+1} = (1-\rho) \cdot \tau(t,u)_k + \rho \cdot \Delta\tau(t,u)$$

dengan perubahan feromon didefinisikan sebagai berikut.

$$\Delta\tau(t,u) = \tau_0.$$

Keterangan:

τ_0 : nilai feromon awal,
$\tau(t,u)_{k+1}$: feromon pada jalur (t,u) setelah iterasi ke-$(k+1)$,
ρ : parameter penguapan feromon $(0 < \rho < 1)$,
$\Delta\tau(t,u)$: intensitas feromon yang ditambahkan pada jalur (t,u).

c. Pembaruan Tingkat Feromon Global

Hanya semut dengan jalur terbaik global yang memperbarui feromon:

$$\tau(t,u)_{k+1} = (1-\alpha) \cdot \tau(t,u)_k + \alpha \cdot \Delta\tau(t,u)$$

dengan menggunakan feromon

$$\Delta\tau(t,u) = \begin{cases} \frac{1}{L_{gb}}, & \text{jika } (t,u) \in \text{tur terpendek dari awal iterasi} \\ 0, & \text{lainnya} \end{cases}$$

Keterangan:
L_{gb} : panjang tur terpendek sejak awal pencarian,
α : parameter bobot feromon $(0 \leq \alpha \leq 1)$.

4 Metode

Penelitian ini menggunakan *Symmetric* TSP (STSP), di mana jarak dari kota *i* ke kota *j* sama dengan jarak dari kota *j* ke kota *i*. Data yang digunakan merupakan data hipotetik dari Jati [11], berupa koordinat Kartesius yang dibangkitkan secara acak menggunakan Microsoft Excel. Terdapat tiga kasus: kasus 1 (20 kota), kasus 2 (40 kota), dan kasus 3 (60 kota).

Penyelesaian TSP dengan metode eksak BnB dilakukan menggunakan *software* LINGO 21.0, sedangkan algoritma ACS diimplementasikan menggunakan Python 3.12.3 pada laptop Acer Swift SF114-34 RAM 4 GB. Hasil ACS dibandingkan dengan solusi BCO [11], ACO [8][12], GWO [13], dan SA [9][14] berdasarkan total jarak dan waktu komputasi.

Parameter ACS mengacu pada Dorigo dan Gambardella [7]: $\rho = 0.1$, $\alpha = 0.1$, $\beta = 2$, dan $q_0 = 0.9$. Nilai feromon awal dihitung dengan $\tau_0 = (nL)^{-1}$, dengan *n* adalah jumlah kota dan *L* adalah solusi eksak. Variasi jumlah iterasi dan semut dilakukan untuk setiap kasus, dan algoritma dijalankan sebanyak 10 kali per kombinasi parameter.

5 Hasil dan Pembahasan

5.1 Implementasi ACS pada Kasus 7 Kota

Sebagai ilustrasi langkah-langkah ACS, algoritma diterapkan pada kasus 7 kota dengan data koordinat yang dibangkitkan secara acak menggunakan Microsoft Excel. Data koordinat dan matriks jarak Euclidean antarkota disajikan pada Tabel 1 dan 2.

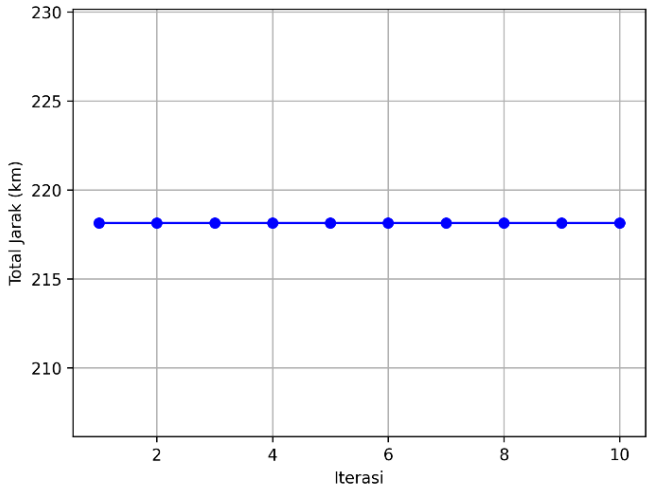
Tabel 1. Data koordinat 7 kota

Kota	x	y
0	13,59	73,93
1	36,14	19,4
2	62,83	86,09
3	70,2	32,7
4	36,15	41,52
5	15,08	65,93
6	56,73	99,71

Tabel 2. Matriks jarak antarkota

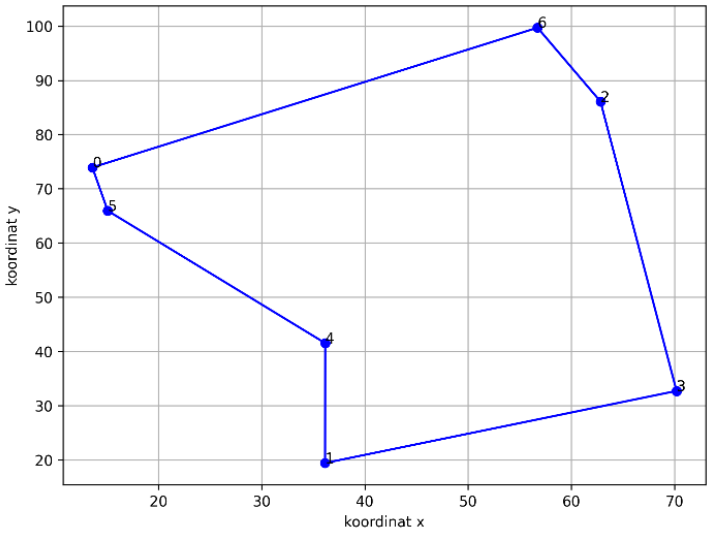
Kota	0	1	2	3	4	5	6
0	0	59,01	50,72	70,03	39,49	8,14	50,26
1	59,01	0	71,83	36,56	22,12	51,07	82,91
2	50,72	71,83	0	53,9	51,95	51,83	14,92
3	70,03	36,56	53,9	0	35,17	64,36	68,35
4	39,49	22,12	51,95	35,17	0	32,25	61,72
5	8,14	51,07	51,83	64,36	32,25	0	53,63
6	50,26	82,91	14,92	68,35	61,72	53,63	0

Algoritma dijalankan selama 10 iterasi. Hubungan antara jumlah iterasi dan total jarak disajikan pada Gambar 1. Terlihat bahwa solusi optimal sudah dicapai sejak iterasi pertama dan tidak mengalami perubahan signifikan hingga iterasi ke-10, menandakan konvergensi yang cepat.



Gambar 1. Grafik hubungan antara jumlah iterasi dan total jarak pada kasus 7 kota.

Solusi akhir ACS adalah rute 0-5-4-1-3-2-6-0 dengan total jarak 218,14 km. Hasil ini identik dengan solusi optimum global yang diperoleh metode eksak BnB menggunakan LINGO 21.0 (waktu komputasi 2 detik). Visualisasi rute ditunjukkan pada Gambar 2.



Gambar 2. Rute TSP untuk kasus 7 kota.

5.2 Hasil Metode Eksak BnB untuk Kasus 20, 40, dan 60 Kota

Metode eksak BnB digunakan untuk memperoleh solusi optimum global sebagai acuan perbandingan. Dari Tabel 3 terlihat bahwa jumlah iterasi dan waktu komputasi BnB meningkat drastis seiring bertambahnya jumlah kota. Pada kasus 3 dengan 60 kota, waktu komputasi mencapai lebih dari 152 jam, yang mengonfirmasi sifat NP-*hard* dari TSP.

Tabel 3. Hasil metode eksak BnB untuk ketiga kasus.

Kasus	Jumlah Kota	Total Jarak (km)	Iterasi BnB	Waktu Komputasi
1	20	368,79	51	4 detik
2	40	480,27	300.949	14 jam 31 menit 11 detik
3	60	540,51	2.315.490	152 jam 59 menit 38 detik

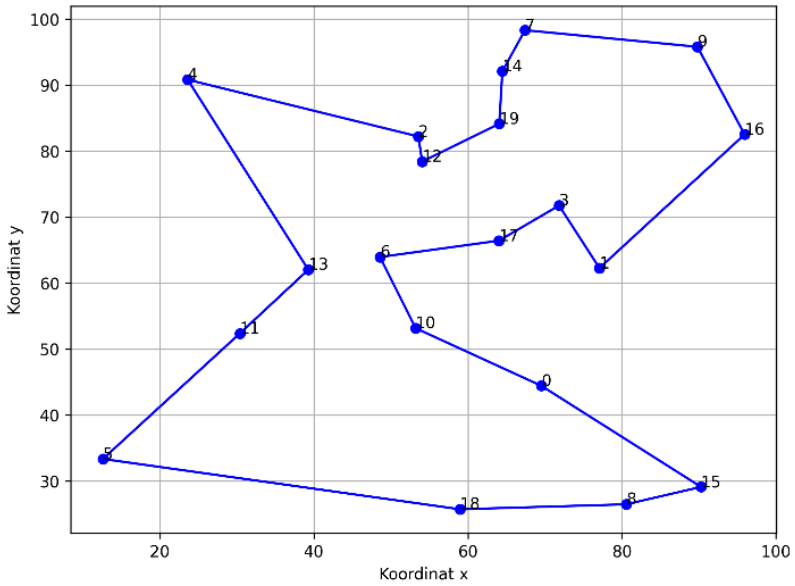
5.3 Pengaruh Jumlah Iterasi dan Semut terhadap Kinerja ACS

Hasil percobaan ACS dengan variasi jumlah iterasi dan semut untuk kasus 1, 2, dan 3 disajikan pada Tabel 4-6.

Tabel 4. Hasil ACS pada kasus 1 (20 kota).

Iterasi	Semut	Total Jarak (km)	Waktu Komputasi
10	5	372,78	0,0289 detik
10	10	371,78	0,0430 detik
10	15	371,78	0,0620 detik
50	5	371,89	0,1058 detik
50	10	371,78	0,2178 detik
50	15	368,79	0,3138 detik
100	5	371,89	0,2098 detik
100	10	368,79	0,4167 detik
100	15	368,79	0,6242 detik

Pada kasus 1, total jarak minimum 368,79 km (sama dengan solusi eksak) pertama kali dicapai pada 50 iterasi dengan 15 semut. Semakin banyak jumlah semut dan iterasi umumnya menghasilkan solusi lebih baik, namun waktu komputasi juga meningkat. Rute perjalanan terbaik kasus 1 disajikan pada Gambar 3.

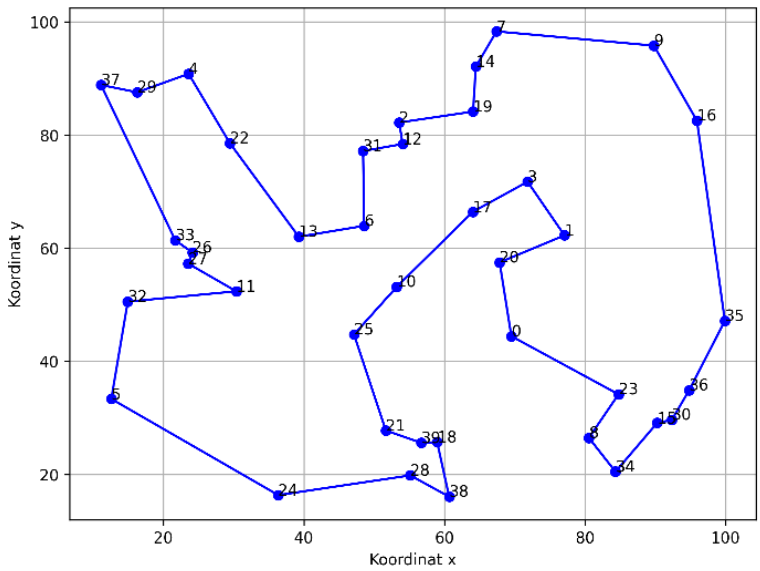


Gambar 3. Rute perjalanan yang dihasilkan algoritma ACS pada kasus 1 (20 kota).

Tabel 5. Hasil ACS pada kasus 2 (40 kota).

Iterasi	Semut	Total Jarak (km)	Waktu Komputasi
50	10	487,09	1,2194 detik
50	20	486,52	2,2664 detik
50	30	486,38	2,8235 detik
100	10	487,09	1,8928 detik
100	20	482,69	3,8335 detik
100	30	480,27	5,7042 detik
150	10	487,09	2,8053 detik
150	20	480,27	5,9287 detik
150	30	480,27	8,6221 detik

Pada kasus 2, total jarak minimum 480,27 km (sama dengan solusi eksak) pertama kali dicapai pada 100 iterasi dengan 30 semut. Rute perjalanan terbaik kasus 2 disajikan pada Gambar 4.

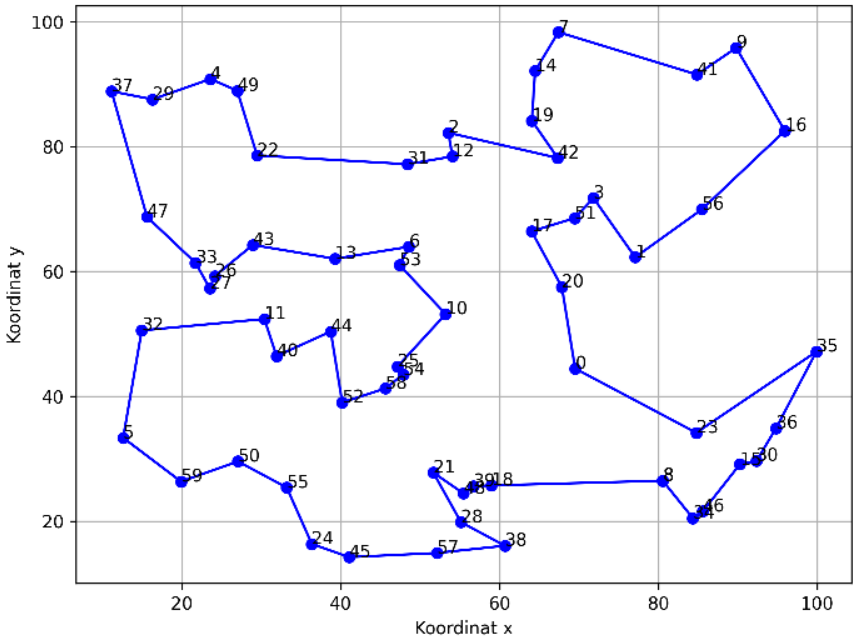


Gambar 4. Rute perjalanan yang dihasilkan algoritma ACS pada kasus 2 (40 kota).

Tabel 6. Hasil ACS pada kasus 3 (60 kota).

Iterasi	Semut	Total Jarak (km)	Waktu Komputasi
150	25	547,43	19,5944 detik
150	35	547,03	27,9867 detik
150	40	548,03	34,3714 detik
200	25	544,47	31,2171 detik
200	35	547,03	44,7942 detik
200	40	542,50	52,2256 detik
250	25	544,46	42,1573 detik
250	35	546,22	53,5295 detik
250	40	542,50	59,9883 detik

Pada kasus 3, total jarak minimum 542,50 km pertama kali dicapai pada 200 iterasi dengan 40 semut. Rute perjalanan terbaik kasus 3 disajikan pada Gambar 5.



Gambar 5. Rute perjalanan yang dihasilkan algoritma ACS pada kasus 3 (60 kota).

5.4 Perbandingan ACS dengan Metode Lainnya

Tabel 7 dan Tabel 8 menyajikan perbandingan waktu komputasi dan total jarak antara ACS dengan metode BnB, BCO, ACO, GWO, dan SA.

Tabel 7. Perbandingan waktu komputasi (jam:menit:detik).

Kasus	BnB	ACS	BCO	ACO	GWO	SA
1	00:00:04	00:00:0,31	00:00:01	00:00:53	00:00:04	00:00:16
2	14:31:11	00:00:05	00:00:11	00:03:35	00:07:51	00:01:13
3	152:59:38	00:00:52	00:00:26	00:08:03	01:43:50	00:09:54

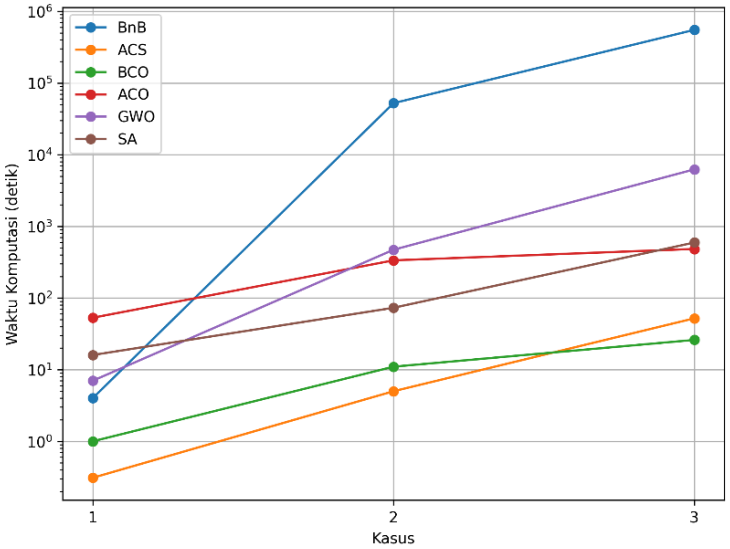
Tabel 8. Perbandingan total jarak (km) dan persentase deviasi (PD) dari solusi eksak.

Kasus	BnB	ACS (PD%)	BCO (PD%)	ACO (PD%)	GWO (PD%)	SA (PD%)
1	368,79	368,79 (0%)	371,89 (0,841%)	368,80 (0,003%)	368,80 (0,003%)	371,89 (0,841%)
2	480,27	480,27 (0%)	494,65 (2,994%)	480,28 (0,002%)	481,88 (0,335%)	495,65 (3,202%)
3	540,51	542,50 (0,368%)	561,57 (3,896%)	557,64 (3,169%)	543,69 (0,588%)	596,84 (10,422%)

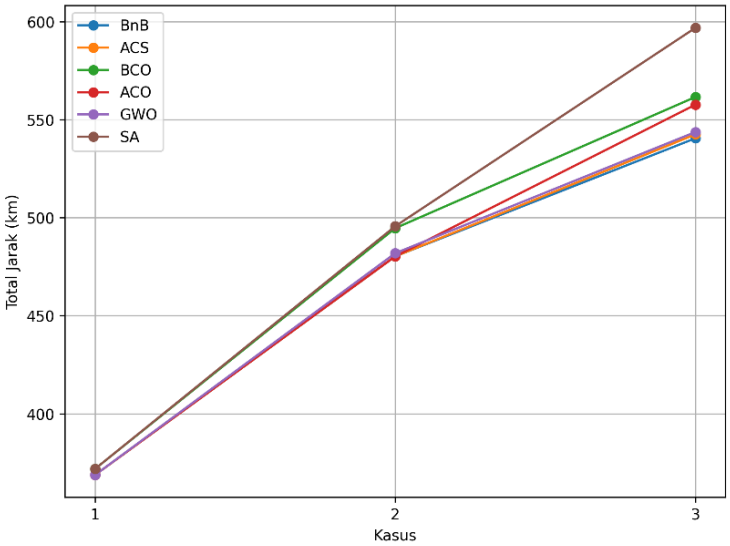
Berdasarkan Tabel 7, waktu komputasi ACS jauh lebih cepat dibandingkan metode eksak BnB pada kasus 2 dan 3. Untuk kasus 2, BnB membutuhkan lebih dari 14 jam

sedangkan ACS hanya 5 detik. Untuk kasus 3, BnB membutuhkan lebih dari 152 jam sedangkan ACS hanya 52 detik. Dibandingkan metode metaheuristik lain, ACS juga lebih cepat kecuali BCO pada kasus 3. Perbandingan tren waktu komputasi seluruh metode disajikan pada Gambar 6.

Berdasarkan Tabel 8, ACS memiliki nilai persentase deviasi yang paling kecil dibandingkan BCO, ACO, GWO, dan SA pada ketiga kasus. Pada kasus 1 dan 2, ACS menghasilkan solusi yang sama persis dengan solusi eksak (PD = 0%). Pada kasus 3, deviasi ACS hanya 0,368%, jauh lebih baik dibandingkan BCO (3,896%), ACO (3,169%), GWO (0,588%), dan SA (10,422%). Hal ini menunjukkan bahwa mekanisme eksplorasi-eksploitasi ACS yang lebih adaptif menghasilkan solusi berkualitas tinggi dengan efisiensi komputasi yang baik. Perbandingan total jarak seluruh metode disajikan pada Gambar 7.



Gambar 6. Perbandingan waktu komputasi untuk setiap metode.



Gambar 7. Perbandingan total jarak untuk setiap metode

6 Simpulan dan Saran

Algoritma ACS berhasil menyelesaikan masalah TSP simetris dengan memanfaatkan mekanisme eksplorasi-eksploitasi dan pembaruan tingkat feromon lokal-global. Penggunaan parameter yang berbeda memengaruhi kualitas solusi dan waktu komputasi: semakin besar jumlah semut dan iterasi cenderung menghasilkan solusi yang lebih baik, meski waktu komputasi meningkat. Namun, kondisi tertentu dapat menyebabkan algoritma terjebak pada solusi lokal optimum.

Dibandingkan dengan BCO, ACO, GWO, dan SA, algoritma ACS menghasilkan total jarak yang paling mendekati solusi eksak BnB dengan waktu komputasi yang kompetitif. Pada kasus 1 dan 2, ACS mencapai solusi optimum global; pada kasus 3, deviasi hanya 0,368%.

Penelitian ini dapat dikembangkan dengan menambahkan kendala seperti *time windows* atau *pick-up and delivery*, menggunakan data dari pustaka TSPLIB, atau menggabungkan ACS dengan algoritma lainnya seperti *Dragonfly Algorithm* atau *Whale Optimization Algorithm*.

Daftar Pustaka

[1] R. N. Puteri, A. W. Widodo, dan I. Cholissodin, Optimasi multiple travelling salesman problem pada pendistribusian air minum menggunakan algoritme particle swarm optimization, *Jurnal Pengembangan Teknologi dan Ilmu Komputer*, vol. 1, no. 9, hlm. 842-848, 2017.

[2] M. V. Andriansyah, R. A. Darajatun, dan D. N. Rinaldi, Optimalisasi pendistribusian dengan metode travelling salesman problem untuk menentukan rute terpendek di PT XYZ, *Tekmapro: Journal of Industrial Engineering and Management*, vol. 16, no. 2, hlm. 84-95, 2021.

[3] K. Auliasari, M. Kertaningtyas, dan D. W. L. Basuki, Optimalisasi rute distribusi produk menggunakan metode travelling salesman problem, *Jurnal Sains Teknologi dan Industri*, vol. 16, no. 1, hlm. 15-23, 2018.

[4] I. Sutoyo, Penerapan algoritma nearest neighbour untuk menyelesaikan travelling salesman problem, *Paradigma*, vol. 20, no. 1, hlm. 101-106, 2018.

[5] C. Dahiya dan S. Sangwan, Literature review on travelling salesman problem, *International Journal of Research*, vol. 5, no. 16, hlm. 1152-1155, 2018.

[6] B. Toaza dan D. Esztergár-Kiss, A review of metaheuristic algorithms for solving TSP-based scheduling optimization problems, *Applied Soft Computing Journal*, vol. 148, hlm. 1-24, 2023.

[7] M. Dorigo dan L. M. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem, *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, hlm. 53-66, 1997.

[8] B. P. Silalahi, N. Fathiah, dan P. T. Supriyo, Use of ant colony optimization algorithm for determining traveling salesman problem routes, *Jurnal Matematika MANTIK*, vol. 5, no. 2, hlm. 100-111, 2019.

[9] B. P. Silalahi, F. Sahara, F. Hanum, dan H. Mayyani, Simulated annealing algorithm for determining travelling salesman problem solution and its comparison with branch and bound method, *JTAM (Jurnal Teori dan Aplikasi Matematika)*, vol. 6, no. 3, hlm. 601-615, 2022.

[10] W. L. Winston, *Operation Research: Applications and Algorithms, Ed. ke-4*. California: Duxbury, 2004.

[11] E. Jati, Optimisasi rute travelling salesman problem menggunakan algoritme bee colony optimization, Skripsi, Institut Pertanian Bogor, Bogor, 2021.

[12] N. Fathiah, Penentuan rute travelling salesman problem menggunakan algoritme ant colony optimization, Skripsi, Institut Pertanian Bogor, Bogor, 2019.

[13] Z. F. Anisa, Penyelesaian travelling salesman problem menggunakan algoritma grey wolf optimization, Skripsi, Institut Pertanian Bogor, Bogor, 2024.

[14] F. Sahara, Penyelesaian travelling salesman problem menggunakan algoritme simulated annealing, Skripsi, Institut Pertanian Bogor, Bogor, 2021.