Improved ant colony-based algorithm for solving traveling salesmen problem

Zhifeng Zheng

School of Engineering, University of California Irvine, Irvine, CA, United States

zhifenz2@uci.edu

Abstract. The purpose of this research is to examine how well the Improved Memetic Continuous Pheromone-based Ant Colony Optimization (ICMPACO) algorithm solves standard Traveling Salesman Problem (TSP) instances in comparison to the basic Ant Colony Optimization (ACO) algorithm and the Improved Ant Colony Optimization (IACO) algorithm. The findings of the computer simulation show that the ICMPACO algorithm has higher optimization capabilities, particularly for the berlin52, eil51, and dantzig42 examples. This superiority is further emphasized through better mean values in comparison to traditional ACO methods. The study also proposes a novel approach, Genetic Ant Colony Optimization (GACO), which integrates ACO with genetic algorithms. Pheromone data is efficiently employed to direct the choice of genetic operation points and maintain the foundational elements within the genetic algorithm. Performance results for various TSP instances reveal the potential of GACO in generating high-quality solutions. These findings suggest that both ICMPACO and GACO methods hold significant promise in addressing complex optimization challenges and could pave the way for further advancements in enhancing optimization algorithm performance.

Keywords: Ant Colony Optimization, Traveling Salesman Problem, optimization.

1. Introduction

The Traveling Salesman Problem (TSP) is a well-known combinatorial optimization problem that has attracted considerable interest in both theoretical and practical research fields, with applications in robotic research and multi-robot coordination [1-3]. At its most basic level, the TSP seeks to determine the most efficient route for a salesman who must visit a collection of cities and return to their starting location. The goal is to reduce the overall distance or expense of the journey, while guaranteeing that every city is visited just once [4]. In reality, the TSP has numerous practical applications across various domains. For instance, it is used in logistics and supply chain management to optimize vehicle routing and minimize transportation costs. Similarly, the TSP has applications in manufacturing for optimizing the sequencing of tasks on assembly lines or the scheduling of machines to reduce production time and costs. Moreover, the TSP finds relevance in areas such as network design, circuit board drilling, and DNA sequencing. In these cases, the problem's core elements remain the same, although the specific objective functions and constraints may vary [5].

Solving the Traveling Salesman Problem (TSP) continues to be challenging due to its NP-hard nature, where finding optimal solutions becomes increasingly difficult as the number of cities grows. Key issues include scalability, as large-scale TSP instances require considerable computational effort, and the need to address real-world complexities like additional constraints, time windows, and varying capacities. Algorithm performance is another challenge, as heuristics and metaheuristics depend on parameter settings and problem characteristics, driving research into adaptive and self-tuning mechanisms. Hybridization of different optimization techniques, can improve solution quality and convergence speed.

Drawing inspiration from the foraging behavior of ants, the Ant Colony Optimization (ACO) algorithm has risen to prominence as a widely used and successful meta-heuristic approach to addressing the Traveling Salesman Problem (TSP). The ACO algorithm emulates the process by which ants leave pheromone trails to convey information and direct fellow colony members towards the most efficient route between their nest and a food source [6]. In the context of TSP, artificial ants construct solutions by traversing graphs representing cities and their interconnected distances, and depositing pheromone trajectories that influence subsequent ants' choices. This iterative process eventually converges to an optimal or near-optimal solution. the success of ACO in solving TSP instances stems from its ability to balance exploration and exploitation, preventing premature convergence to suboptimal solutions [7]. With the increasing complexity of large-scale optimization problems, even advanced ACO algorithms encounter certain constraints when tackling these demanding challenges. They may exhibit slow convergence speeds and a propensity to become trapped in local optima. Tackling these issues and improving the performance of ACO algorithms in large-scale optimization problems have become critical challenges for researchers and practitioners [8].

Numerous ACO variants have emerged over the years, generally falling into three categories: algorithm parameter control, pheromone updating strategies, and integration with auxiliary search operators. To counteract premature convergence, some approaches propose updating pheromone concentrations based solely on the best-performing ants' trails while restricting pheromone levels to a finite range for optimal performance. Others have suggested an algorithm combining the Ant Colony System (ACS) with the Taguchi method for solving the TSP, leveraging Taguchi's parameter optimization technique to enhance the algorithm's optimization capabilities. Moreover, an improved ACO variant conducts multiple explorations within a single iteration to expedite the discovery of optimal solutions and dynamically adjusts parameters to prevent ants from revisiting paths. Despite significant advancements in search performance, ACO algorithms still grapple with limitations, including premature convergence, susceptibility to local optima, and subpar performance in high-dimensional, complex problems [9].

To tackle the issue of premature convergence and susceptibility to local optima in the ACO algorithm, a pheromone update and weighted combination of transfer probabilities utilizing fractional order differences is suggested. Additionally, a hybrid approach, integrating genetic algorithms with the ACO algorithm, is proposed to enhance the optimality finding capability and stability when addressing more intricate TSP problems. Computer simulation results indicate that, compared to the traditional ACO algorithm, the optimized parameters and combination with other algorithms significantly improve the ACO algorithm's speed, optimization ability, and stability when solving TSP problems.

2. Problem formulation

2.1. ACO algorithm

The ACO is mainly composed of transition rules and pheromone update rules, for transition rules the expression is.

$$\tau P_{ij}^{k}(t) = (\tau_{ij}(t)^{\alpha} * \eta_{ij}^{\beta}) * \Sigma(\tau_{il}(t)^{\alpha} * \eta_{il}^{\beta})^{-1} \tag{1}$$

where $P_{ij}^{k}(t)$ is the probability of ant k moving from city i to city j at time t. $\tau_{ij}(t)$ is the pheromone trail intensity between city i and city j at time t. η_{ij} is the heuristic information, typically the inverse of the distance between city i and city j. α and β are parameters controlling the relative importance of pheromone trails and heuristic information.The denominator represents the sum of all possible edges' attractiveness from city i to all other unvisited cities l.

After ants complete their tours, the pheromone trails on the edges are updated to reinforce the paths taken by the ants. The pheromone update formula is:

$$\tau(r,u) = (1-\rho)\tau(r,s) + \sum_{k=1}^{m} \Delta\tau_k(r,s) \tag{2}$$

where ρ ($0 < \rho < 1$) represents the pheromone trail evaporation rate. $\Delta\tau_k(r,s)$ refers to the quantity of pheromone trail deposited by ant k.on the edge connecting nodes r and $s$during the time interval between t and $\Delta t + t$ within its tour. The following explanation elaborates on this concept:

$$\Delta\tau_k(r,s) = \begin{cases} \frac{Q}{L_k} & (r,s) \in \pi_k \\ 0 & otherwise \end{cases} \tag{3}$$

2.2. Genetic algorithm

The Genetic Algorithm (GA) has proven highly effective in addressing machine learning and optimization challenges. To tackle a problem, the GA employs a population of individuals, often referred to as strings or chromosomes, and modifies this population probabilistically using genetic operators like selection, crossover, and mutation [10].

3. Method

3.1. ICMPACO

The The natural process of co-evolution served as the basis for the development of the Co-evolutionary algorithm, which uses the principles of decomposition and coordination to partition complex optimization challenges into a number of interconnected sub-challenges for optimization and coordination. This paper develops a novel Multiple Swarm Co-evolutionary Ant Colony Optimization (ICMPACO) algorithm for solving large-scale optimization problems by incorporating a number of different swarm strategies, co-evolutionary mechanisms, pheromone update approaches, and pheromone diffusion techniques into the ACO algorithm [8]. In the ACO method, just one species of ant is used to develop solutions. These solutions can be regulated by the parameters for colony size, selection, and convergence. Several swarming tactics are utilized, with the ants being separated into elite and regular categories. To increase the rate at which ACO converges, Elite Ants retrieve information from a solution library and generate solutions by employing Gaussian kernel functions and probabilistic selection procedures, in addition to the parameters that are specific to them. Normal ants, on the other hand, make use of a single Gaussian function and dimension values that are on average, generating new solutions at a slower rate in order to avoid becoming trapped in local optima [8]. The equations are written as follows:

$$f_N^i(x) = \frac{1}{\sigma_{i,N}\sqrt{2\pi}} e^{-\frac{(x-\mu_{i,N})^2}{2\sigma_{i,N}^2}} \tag{4}$$

$$\mu_{i,N} = \sum_{k=1}^{K} s_{i,k} \tag{5}$$

$$\sigma_{i,N} = \xi_N \sum_{e=1}^{K} \frac{|s_{i,e} - \overline{s_i}|}{K-1} \tag{6}$$

where $f_N^i(x)$ represents the Gaussian function for normal ants in the ith dimension, with $\mu_{i,N}$ as its sample value and $\sigma_{i,N}$ as the calculated standard deviation. s_i is the average value of solutions in the ith dimension, while ξ_N is a constant controlling the convergence rate of common ants. Consequently, common ants can efficiently expand the search range and boost global search capabilities. Pheromone update is crucial in the optimization process [8]. New pheromone updating tactics and diffusion mechanisms have been presented in order to improve the performance of the ACO algorithm when applied to the solution of difficult optimization problems. Among them are strategies for both local and global pheromone updates. The equation for a local pheromone update is as follows:

$$\tau_{x,y}^{(i)} = (1-\rho_L)\tau_{x,y}^{(i)} + \rho_L \Delta\tau_0^{(i)} \tag{7}$$

where $\rho_L \in (0,1)$ is the local pheromone evaporation coefficient, and $1-\rho_L$ represents the pheromone residue factor. is the initial pheromone value, with $\tau_0^{(i)}$ as a small negative number when the node value is 1, and $\tau_0^{(i)}$ equal to 0 when the node value is 0. So the expression for global pheromone update strategies is

$$\tau_{x,y}^{(i)} = (1-\rho_G)\tau_{x,y}^{(i)} + \rho_G \Delta\tau_G^{(i)} \tag{8}$$

$$\Delta\tau_G^{(i)} = \begin{cases} F_G^{(i)}, & (x,j) \in \text{Global optimal solution} \\ F_I^{(i)}, & (x,j) \in \text{Iterative optimal solution} \\ 0, & otherwise \end{cases} \tag{9}$$

where $\rho_G \in (0,1)$ denotes the global pheromone evaporation coefficient, while $1-\rho_G$ represents the pheromone residue factor. The solution denoted by the notation F_G refers to the global optimal solution, and $F_I^{(i)}$ signifies the iterative optimal solution. The coevolutionary mechanism, a recent evolutionary concept based on coevolution theory, acknowledges organism diversity, and stresses the interdependence between organisms and their environment during evolution. This mechanism applies coevolution theory to establish competitive or cooperative relationships among multiple populations, enhancing optimization performance through their interactions [11]. By emphasizing the influence of different subpopulations on each other and their coevolution, the coevolution mechanism is incorporated into the ACO algorithm, enabling information exchange between distinct subpopulations.

3.2. GACO

The GACO approach incorporates two genetic operations: crossover and mutation. To prevent the disruption of useful patterns or building blocks in the genetic algorithm by these operations, a Linkage Strength-based learning mechanism is employed. In the beginning of an operation known as a two-point crossover, the genetic algorithm analyzes the pheromone value that exists in both cities to determine the linkage strength that exists among nearby genes on the parent chromosome. Subsequently, it selects crossover and mutation points based on linkage strength through a roulette method. Then, a random limit is constructed that lies between the minimum and maximum possible values for the chromosomal linkage strength. The linkage strength between exchanged chromosome genes is categorized as either strong or weak. Strong links form subroutes, while weak links represent individual cities. Future generations of chromosomes are constructed accordingly [12].

In addressing the TSP problem, both ICMPACO and GACO have demonstrated their effectiveness in discovering optimal solutions. However, the specific performance of each algorithm may depend on the size and complexity of the problem being solved. In summary, the relationship between ICMPACO and GACO stems from their utilization as ACO algorithms to identify optimal solutions for the TSP problem, achieved by modifying the pheromone trail according to the actions of artificial ants.

4. Results

To demonstrate that the optimized ACO algorithm outperforms the traditional ACO algorithm in solving the TPS problem, computer simulations are used here to demonstrate. In this study, the efficacy of the ICMPACO method is assessed using eight Traveling Salesman Problems (TSPs) sourced from the TSPLIB standard library. To determine the distance between any two cities, Euclidean distance is employed, in line with the TSPLIB's attributes, and the resulting values are rounded to the nearest whole number.To determine how well the proposed ICMPACO algorithm performs in terms of enhancement, it is compared both to the original Ant Colony Optimization (ACO) algorithm and to an improved version of ACO that includes group intelligent local search. This is done so that we can evaluate the effectiveness of the ICMPACO algorithm (IACO).

In order to evaluate how well the proposed ICMPACO performs in terms of optimization, it is compared with the standard Ant Colony Optimization (ACO) method as well as an upgraded version of

ACO called an improved ACO that includes group intelligent local search (IACO) [8] with the parameters shown in Table 1.

Table 1. values for parameters.

Algorithms	*ACO*	*IACO*	*IMPACO*
Ants(K)	30	30	30
Pheromone factor(α)	1	1	1
Heuristic factor(β)	0.5	0.5	0.5
Volatility coefficient(ρ)	0.1	0.1	0.1
Pheromone amount(Q)	100	100	100
centration($\tau_{ij}(0)$)	1.5	1.5	1.5
Maximum iterations(T)	200	200	200

Table 2. Experimental results.

Instances	Algorithms	Optimal value	Maximum value	Minimum value	Average value	Variance
dantzig42	ACO	699	726.2402	707.7596	718.5473	9.2403
	IACO		737.1451	704.6353	717.58993	16.2549
	ICMPACO		718.8354	703.1199	711.02192	7.8578
eil51	ACO	426	455.8169	443.3749	449.91115	6.221
	IACO		452.6667	440.6427	446.17787	6.012
	ICMPACO		439.9814	429.8871	435.24398	5.0472
berlin52	ACO	7542	7757.4	7663.6	7687.21	46.9
	IACO		7681.5	7589	7622.4	66.25
	ICMPACO		7613.7	7548.6	7621.36	37,55
cil101	ACO	629	708.2028	683.7806	694.7517	12.2111
	IACO		702.5375	680.3677	692.7951	11.0849
	ICMPACO		686.246	668.236	677.4336	9.0050
pr107	ACO	44303	46585	46124	46414.6	230.5
	IACO		46292	45690	46050.3	301
	ICMPACO		46103	45649	45970.6	227
ch130	ACO	6110	6473.2	6311.2	6399.8	81
	IACO		6467.8	6307.5	6384.83	80.15
	ICMPACO		6273.5	6183.4	6235.95	45.05
kroA200	ACO	29368	37062	32041	33763	132.5
	IACO		36849	31974	32871	127.3
	ICMPACO		36134	31267	32086	136.4
rat783	ACO	8806	11508	9932	10791	70.4
	IACO		11352	9453	9804	82.1
	ICMPACO		10891	9229	9672	85.3

Table 2 displays the results of experiments that were conducted in order to solve classic TSP examples such as dantzig42, eil51, berlin52, eil101, pr107, ch130, kroA200, and rat783 utilizing the fundamental ACO method, the IACO algorithm, and the suggested ICMPACO algorithm. In considering the results presented in Table 2, it is clear that the optimization overall quality of the three algorithms and their efficacy in dealing with the TSP situations are comparable. The ICMPACO method consistently produces the best optimization values for those common TSP cases. This is especially true for berlin52, eil51, and dantzig42. When compared to the ideal values of 7542, 426, and 699, the results that are produced by the ICMPACO method are 7548.6, 429.8871, and 703.1199, respectively. These numbers are near to being optimal. It would appear from this that the ICMPACO method has a greater capacity for optimization than either the ACO or IACO algorithm when it comes to the solution of these common TSP instances. In addition to this, the ICMPACO algorithm produces the highest average value in terms of the mean value, demonstrating its more pronounced advantage in terms of optimization performance [3].

Similarly, to prove the validity of GACO, the TSP problem was selected as shown in Table 3, with the parameters shown in Table 4.

Table 3. The problem of choice.

Name	Size (eities)	Best-known cost
ulysses16	16	74.1088
gr24	24	1272
Ei151	51	429.9835
st70	70	678.5973
cil76	76	545.3873
kroA100	100	21285

Table 4. Selected parameters.

Psize	P.crossover	Pmutation	Max_num_trials	β	γ	p	q	ρ
50	0.85	0.05	1,000,000	2	0.1	0.1	0.90	0.1

Table 5. Experimental results.

Problem	Best-tour length length	RPD (%)	Best-tour length length	RPD (%)
ulysses16	73.9877	- 0.0016	74.0567	-0.0931
gr24	1272	0	1274.5	0.2
E151	429.9835	0	431.1503	0.3
st70	678.5973	0	682.7469	0.6
eil76	545.3873	0	548.3518	0.5
kroA100	21285	0	21723	2

In Table 5, "cost" stands for the distance traveled to reach the answer, whereas "opt" indicates the optimal cost that is currently known. It is important to note that the Relative Percentage Difference (RPD) for the experiments conducted on Ulysses16 is negative, indicating that the optimal tour identified by our solution is shorter than the previously known optimal solution [7].

Throughout the course of the simulation tests, a number of different parameter values for individual functions were investigated and tinkered with in order to ascertain which provided the best starting point for these parameters. The ICMPACO method, according to experimental results, consistently produces the best metaheuristic values for those common Travelling salesman problem instances, particularly berlin52, eil51, and dantzig42. ICMPACO beats both the fundamental ACO and IACO algorithms in terms of optimization capability for these common TSP situations, achieving near-optimal results in contrast to ideal values. Moreover, ICMPACO offers the best average value in terms of mean value, highlighting its specific performance advantage in optimization. Overall, the results of this work emphasize ICMPACO's capacity to handle challenging optimization problems by emphasizing its better optimization skills over conventional ACO algorithms in tackling TSP problems. The suggested approach for GACO combines genetic algorithms and ant colony optimization, with each member of the GA acting as an ant in the ACO. The identification of the genetic algorithm's basic components and the direction of the choice of genetic action locations are both successfully accomplished using pheromone information. By actively guiding the methodology and constricting the TSP search area, the ACO-based building block learning mechanism enables the GA to avoid frequent disturbances to its basic components caused by crossover and mutation operations. Data collected for several TSP scenarios demonstrate this method's capacity to deliver high-caliber results. The underlying ideas for such approach may pave the way for novel methods to improve GA performance, allowing for more precise and effective genetic manipulation.

5. Conclusion

In conclusion, the purpose of this research was to use computer simulations to demonstrate that the optimized version of the ACO algorithm, known as ICMPACO, is superior to the classic version of the ACO algorithm when it comes to addressing TSP issues. Eight different TSP instances taken from the TSPLIB standard library are used in the analysis to see how successful the ICMPACO technique is. In order to determine the distances between cities, the Euclidean distance is employed, which is consistent with the features of the TSPLIB. The values that are calculated using this distance are then rounded to the nearest whole number. Comparisons are made between the improvement capabilities of the proposed ICMPACO algorithm and those of the traditional ACO as well as an improved version of the ACO that makes use of group intelligent local search (IACO).

References

[1] Chen J and Dames P 2020 Distributed multi-target search and tracking using a coordinated team of ground and aerial robots In Robotics science and systems
[2] Chen J, Xie Z and Dames P 2022 The semantic PHD filter for multi-class target tracking: from theory to practice *Robotics and Autonomous Systems* 149 103947
[3] Chen J and Dames P 2022 Multi-class target tracking using the semantic phd filter In Robotics Research: The 19th International Symposium ISRR p 526-541
[4] Shi K, Zhang H, Zhang Z and Zhou X 2020 The algorithm of terminal logistics path planning based on TSP problem International Conference on Artificial Intelligence and Computer Engineering (ICAICE) pp. 130-133
[5] L Hoffman, M Padberg and G Rinaldi 2013 Traveling salesman problem *Encyclopedia of operations research and management science* pp.1573-1578.
[6] Dorigo M, Maniezzo V and Colorni A 1996 Ant system: optimization by a colony of cooperating agents *IEEE Transactions on Systems Man and Cybernetics Part B (Cybernetics)* 29–41
[7] Dahan F, Hindi K, Mathkour H and AlSalman H 2019 Dynamic flying ant colony optimization (DFACO) for solving the traveling salesman problem *Sensors* 19(8):1837
[8] Gong D and Ruan X 2004 A hybrid approach of GA and ACO for TSP Fifth World Congress on Intelligent Control and Automation pp. 2068-2072 Vol.3
[9] Guo P, Wang X and Han Y 2010 The Enhanced genetic algorithms for the optimization design 2010 3rd international conference pn Biomedical Engineering and Infromatics pp. 2990-2994
[10] Deng W, Xu J and Zhao H 2019 An improved ant colony optimization algorithm based on hybrid strategies for scheduling problem in *IEEE Access* 20281-20292
[11] Zeng W and Chow M 2012 Optimal tradeoff between performance and security in networked control systems based on coevolutionary algorithms in *IEEE Transactions on Industrial Electronics* 3016-3025
[12] Gong X, Rong Z, Gao T, Pu Y and Wang J 2019 An improved ant colony optimization algorithm based on fractional order memory for traveling salesman problems Symposium Series on Computational Intelligence (SSCI) pp. 1516-1522