Acadlore Transactions on AI and Machine Learning
https://www.acadlore.com/journals/ATAIML

A Comprehensive Review of Ant Colony Optimization in Swarm Intelligence for Complex Problem Solving

Batool Abdulsatar Abdulghani[1*], Mohammed Abdulsattar Abdulghani[2]

[1] Department of Energy Engineering, Duhok Polytechnic University, 42001 Duhok, Iraq
[2] Department of Computer Science, College of Education for Pure Sciences, University of Hamdaniya, 41002 Mosul, Iraq

[*] Correspondence: Batool Abdulsatar Abdulghani (batool.abdulghani@dpu.edu.krd)

Received: 09-18-2024 **Revised:** 10-27-2024 **Accepted:** 11-03-2024

Citation: B. A. Abdulghani and M. A. Abdulghani, "A comprehensive review of Ant Colony Optimization in swarm intelligence for complex problem solving," *Acadlore Trans. Mach. Learn.*, vol. 3, no. 4, pp. 214–224, 2024. https://doi.org/10.56578/ataiml030403.

Abstract: Swarm intelligence (SI) has emerged as a transformative approach in solving complex optimization problems by drawing inspiration from collective behaviors observed in nature, particularly among social animals and insects. Ant Colony Optimization (ACO), a prominent subclass of SI algorithms, models the foraging behavior of ant colonies to address a range of challenging combinatorial problems. Originally introduced in 1992 for the Traveling Salesman Problem (TSP), ACO employs artificial pheromone trails and heuristic information to probabilistically guide solution construction. The artificial ants within ACO algorithms engage in a stochastic search process, iteratively refining solutions through the deposition and evaporation of pheromone levels based on previous search experiences. This review synthesizes the extensive body of research that has since advanced ACO from its initial ant system (AS) model to sophisticated algorithmic variants. These advances have both significantly enhanced ACO's practical performance across various application domains and contributed to a deeper theoretical understanding of its mechanics. The focus of this study is placed on the behavioral foundations of ACO, as well as on the metaheuristic frameworks that enable its versatility and robustness in handling large-scale, computationally intensive tasks. Additionally, this study highlights current limitations and potential areas for future exploration within ACO, aiming to facilitate a comprehensive understanding of this dynamic field of swarm-based optimization.

Keywords: Ant Colony Optimization; Metaheuristics; Swarm intelligence; Pheromone; Combinatorial

1 Introduction

The use of optimization algorithms involves finding the best possible solution to a given problem. The purpose of an optimization algorithm is to find the optimal solution that minimizes or maximizes a given objective function [1]. The best-known SI algorithms are those that are derived from the group behaviors of organisms including ants, bees, wasps, termites, fish, and birds. SI algorithms are frequently employed to address complex optimization issues [2]. It was from the social behavior of competing animals for food that SI originated. SI-based algorithms are characterized by their particles, which are basic agents that move in the decision space and collaborate through indirect communication. ACO and Particle Swarm Optimization (PSO) are two of the best SI-inspired optimization techniques. Complex combinatorial optimization (CO) problems are solved using the metaheuristic ACO [3]. Inspired by the pheromone-using behavior of biological ants, the fundamental algorithm of ACO is built on the laying of pheromone trails. ACO is based on simple agents, such as artificial ants that communicate indirectly with one another inside a colony via artificial pheromones, much like the biological version. ACO is predicated on indirect communication caused by artificial pheromone bands among a colony of simple agents known as artificial ants. One of the main ways that the ACO algorithm differs from other construction heuristics is that ants use traces of pheromones, which function as dispersed and digital information, to construct solutions to the problem at hand and adjust to it during the algorithm's execution. AS [4–6] is one of the early examples of this algorithm, along with a notable application example of TSP [7]. Despite AS's encouraging performance, it is insufficient to match the most sophisticated TSP algorithms. Nevertheless, it is crucial in promoting additional research into the two algorithmic variations that offer noticeably more processing power and applications. Algorithms that follow the ACO metaheuristic are known as

ACO algorithms. Since the initial ACO algorithm was introduced in 1992, hundreds of researchers have become interested in the topic. At the moment, ACO is thought to be a highly developed metaheuristic, having at its disposal a substantial amount of theoretical and experimental research findings.

2 Historical Development of ACO

This section discusses the evolution of ACO algorithms over the years, including notable advancements and modifications made to ACO methods that have enhanced their performance and applicability. Dorigo [5] presented the ACO algorithm in 1992 as a SI field approach in his PhD thesis, which was used as a nature-inspired metaheuristic for solving hard CO problems. Dorigo and Gambardella [8] published the Ant Colony System (ACS), a developed algorithm, subsequent in 1997. Asaad and Abdulnabi [9] examined the enhancement of the Ant Colony Optimization (ACO) algorithm for solving the Traveling Salesman Problem (TSP). Over the past 31 years from 1992 to 2024, AS methods, especially ACO, have been modified to solve many optimization problems in real-life situations.

Janakiraman [10] proposed a Hybrid Ant Colony and Artificial Bee Colony Optimization technique for effective cluster head selection (HACO-ABC-CHS) in Internet of Things (IoT) networks. The HACO-ABC-CHS scheme aims to eliminate the limitations of ACO and Artificial Bee Colony (ABC) algorithms by integrating their exploration and exploitation abilities. It partitions the exploitation process into two levels using employee bee phases for primary exploitation. The proposed technique is evaluated based on alive nodes, residual energy, dead nodes, and throughput. Experimental results show that it outperforms other clustering techniques. Tran et al. [11] centered on the importance of Maintenance, Repair, and Overhaul (MRO) in the remanufacturing business within the framework of Industry 4.0. In addition to discussing the issues and features of task scheduling for MRO parts, they also investigated the application of predictive maintenance, dependability techniques, and a unique model for inspection scheduling of welded components and parts made by additive manufacturing. In order to address the scheduling optimization of MRO procedures, the ACO algorithm was used, aiming to minimize the overall amount of time spent on scheduling as well as the total amount of tardiness for all jobs. The algorithm also has dynamic scheduling capabilities to adapt to changes on the shop floor.

In addition, Zhu et al. [12] discussed the comparison of different ACO algorithms for solving TSP instances. They employed an algorithm known as Multiple Colonies ACO based on Pearson Correlation Coefficient (PCCACO), which is designed to mitigate the problems associated with ACO, particularly on large-scale TSPs, such as slipping into local optima and having low variety. The PCCACO algorithm was compared with ACS and Max-Min Ant System (MMAS) algorithms, showing superior performance in all scale TSP instances and metrics. They also explained the use of Pearson correlation coefficient to measure the similarity of different colonies and exchange information between them. Additionally, the pheromone update rules for MMAS and the limitations imposed to avoid fast convergence and stagnation were described. The experimental results and comparisons were presented, highlighting the convergence speed and solution quality of PCCACO in different TSP instances. Overall, the approach provides insights into the development and evaluation of PCCACO as an effective approach for solving large-scale TSPs. An improved ACO approach was proposed by Wang et al. [13] in order to address the issue of the unmanned surface vehicle (USV) collision avoidance planning algorithm based on the Collision Regulations at Sea (COLREGS). They presented a motion velocity model for the USV to establish collision avoidance areas and ensure safety for both the USV and dynamic obstacles. The approach simulated and verified collision avoidance planning for multiple dynamic known obstacles based on COLREGS. Additionally, they referenced various path-planning algorithms for ships and unmanned vehicles, such as the ACO algorithm, and their applications in robotic path planning and obstacle avoidance.

Gao [14] addressed the drawbacks of conventional ACO by using a new ACO algorithm to solve CO problems, specifically the TSP. By mixing pairs of searching ants to promote solution diversification, the new method seeks to increase the size of the ants' search field and diversify the searched solutions. Using 20 typical TSP examples, the study compared the performance of the novel ACO algorithm with that of classic ACO and several state-of-the-art algorithms. Based on the results, it can be concluded that the new algorithm is a good approach to solving the TSP because its solutions are generally better than those of most existing methods. In contrast to conventional ACO and some cutting-edge algorithms, the novel ACO method showed improved computing stability, faster computation speed, and higher computing precision. In addition, it is observed that the computing efficiency of the new ACO method increases with the problem's complexity. Although more testing, particularly for large-scale TSPs and other CO problems, is noted as the next objective, it is admitted that the new approach has only been tried on relatively small TSPs.

Liu and Cao [15] employed a novel ACO algorithm called Levy ACO that incorporates the Levy flight mechanism. Levy flight is inspired by the movement patterns of animals and can help address the exploration-exploitation dilemma in reinforcement learning algorithms like ACO. In Levy ACO, the Levy flight mechanism alters the selection probabilities of candidate solutions to encourage exploring less favored solutions and improve diversity. Computational experiments on TSP instances show that Levy ACO finds best-known solutions with fewer iterations on average compared to the max-min ACO algorithm and other recent TSP solvers. An upgraded ant colony algorithm for multi-factor path planning was presented by Wang et al. [16] to address the issues of low path quality and a single evaluation factor in path planning using the traditional ant colony approach. The goal of the algorithm is to avoid obstacles in complicated static environments. The traditional ant colony algorithm only considers distance as a factor and does not account for other environmental influences. The improved algorithm introduced a fuzzy planner to comprehensively evaluate factors like geological hazards, land use, slopes, and geology. The ant colony algorithm's pheromone updating formula and probability selection formula were also optimized. In simulation experiments, the improved algorithm produced a smoother path with lower negative environmental impacts compared to the traditional algorithm.

In addition, an improved ACO algorithm called ACO Context Aware (ACOCA) was proposed by Liang et al. [17] for tourism route planning to solve the problem of one-sided pursuit of the shortest distance. Contextual information was introduced, like weather and comfort levels of scenic spots, to decide how pheromone levels are updated. They aimed to balance route distance and user comfort. The algorithm was evaluated on different scenic spot datasets and showed improvements over basic ACO in distance, convergence time and user comfort ratio. ACO was described as being applied to problems like routing, scheduling and TSP. Skackauskas et al. [18] presented a new method called Dynamic Impact as an extension to the ACO algorithm. Dynamic Impact is intended to assist ACO in resolving optimization issues in which the fitness and resource consumption connection depends on other aspects of the solution. The Multi-Dimensional Knapsack Problem (MKP) and the Microchip Manufacturing Plant Production Floor Optimization (MMPPFO) problem were used to test the approach. Both problems have objectives that depend on collections of components rather than individual components. The Dynamic Impact method calculated the participation of each edge to the fitness based on its remaining resource consumption. The approach aimed to guide the ants towards edges that improve fitness the most relative to their resource usage. The method improved the solution quality of ACO on both test problems compared to not using Dynamic Impact.

Skinderowicz [19] provided a brand-new ACO algorithm for big TSP situations termed Focused ACO (FACO). A more effective local search is made possible by the FACO, which regulates the quantity of additional edges (components) added to ant-constructed solutions in comparison to source solutions. To lessen computing complexity, this strategy was coupled with other methods like candidate and backup lists. According to an experimental investigation, FACO outperforms current ACO techniques in finding high-quality solutions to TSP instances with more than 100,000 nodes. Morin et al. [20] presented an ACO algorithm called ant search path with visibility (ASPV) for solving the optimal search path problem with visibility (OSPV). The OSPV is an NP-hard path planning problem that occurs in search and rescue operations. It involves finding search paths that maximize the probability of detecting a moving search object, given constraints on search resources and visibility. The ASPV algorithm was evaluated on different problem instances and configurations. Results showed that the best ASPV variant outperforms a greedy heuristic and MILP solver, especially on larger instances. Problem-specific pheromone initialization, iteration-best updates, restarts, and boosting were key components of high-performing variants. Frías et al. [21] proposed two hybrid algorithms called Free Ant and Restricted Ant to solve the Energy Minimizing Vehicle Routing Problem (EMVRP). The algorithms combine ACO with clustering techniques like K-Medoids and K-Means. In Free Ant, clusters guide the initialization of pheromone values and the ant's movement between clusters. In Restricted Ant, clusters restrict the ant's movement within a group. The algorithms were tested on instances from CVRPLIB and compared to results from other works. The Free Ant algorithms generally find better solutions but the Restricted Ant is faster, making it better for larger problems. Limitations include dependency on parameter tuning and lack of benchmark problem instances for the EMVRP model.

A new network alignment approach called AntNetAlign, which is based on ACO, was presented by Corominas et al. [22]. Any pairwise node similarity information could be used by the algorithm to direct the alignment creation process. It integrates a local improvement measure that considers the present solution construction state with the global similarity metric. Real-world examples from domains such as social networks and Protein-Protein Interaction (PPI) networks were used to assess the method. The results demonstrated that AntNetAlign achieves near-optimal outcomes when aligning networks with themselves and surpasses other approaches in terms of the S3 quality score. Additionally, it demonstrated a strong tolerance to network noise. A 3D surface laser scanning path planning method using fuzzy logic and adaptive ACO with subpopulation was presented by Song et al. [23]. In order to determine the ideal measurement attitude of the laser probe, three coordinate systems were created using a four-coordinate measuring device as a basis. As the optimization goal for the fuzzy logic and adaptive ACO (SFACO) method, a nominal distance matrix was created using the machine axes' data under this ideal attitude. To improve performance, the algorithm used dynamic 3-opt neighborhood structure, adaptive heuristic factors, and sub-population techniques. Its efficacy in determining the best scanning path was demonstrated when it was used to 3D laser scanning path planning for complicated surfaces after it had been tested using TSP benchmark datasets.

Sgarro and Grilli [24] suggested utilizing an ACO algorithm (ACO-CPP) to solve the Chinese Postman Problem (CPP). The CPP required navigating every edge in a linked, undirected graph by taking the shortest path possible. The odd nodes in the graph were first found by the ACO-CPP method, which then created a matrix of the shortest pathways connecting these odd nodes. In order to reduce the overall cost of the graph augmentation, the best combination of these shortest paths was found via an ACO search. Through three trials, the ACO-CPP algorithm's performance was assessed and contrasted with that of a recursive and genetic algorithm (GA). The outcomes showed that ACO-CPP was effective and reliable, especially when dealing with issues with many possible solutions. Table 1 summarizes the advancements in ACO algorithms, highlighting their approaches to pheromone trail updating and their performance impacts.

Table 1. Developments in ACO algorithms

Reference	Method/Algorithm	Problem Addressed	Pheromone Trail Updating Approach	Impact on Performance
[5]	ACO	CO problems	Basic pheromone deposit, evaporation for decay	Foundational model; suffers from stagnation in complex problems
[8]	ACS	Path optimization	Global and local pheromone updating	Improved convergence speed and solution quality over basic ACO
[9]	ACO with Great Deluge and 2-Opt (inside ACO)	TSP	Enhances pheromone trails by refining path selections iteratively within ACO	Achieves near-optimal solutions with improved path accuracy, but slower due to added processing
[10]	HACO-ABC-CHS	Cluster head selection in IoT networks	Integrates ACO with ABC; two-level exploitation phases	Outperforms other clustering techniques; enhanced network performance metrics
[11]	ACO for MRO scheduling	Scheduling optimization for MRO processes	Dynamic pheromone updating for scheduling tasks	Minimizes overall scheduling time and tardiness; adapts to shop floor changes
[12]	PCCACO	TSP	Uses Pearson correlation for pheromone sharing; distinct update rules	Superior performance in large-scale TSPs; improves solution diversity and convergence speed
[13]	ACO for USV collision avoidance	Collision avoidance planning for USVs	Pheromone updates based on collision avoidance areas	Effective path planning in dynamic environments; ensures safety for USVs
[14]	Improved ACO	TSP	Pheromone diversification through mixed pairs of searching ants	Better solutions and computing stability, faster computation speed and improved precision
[15]	Levy ACO	TSP	Incorporates Levy flight mechanism for selection probabilities	Finds best-known solutions with fewer iterations; enhances exploration efficiency
[16]	Multi-Factor Path Planning	Path planning in complex static environments	Optimizes pheromone updating based on multiple factors	Generates smoother paths with lower negative environmental impacts
[17]	ACOCA	Tourism route planning	Updates pheromone levels based on contextual information	Balances route distance and user comfort; improves overall satisfaction
[18]	Dynamic Impact ACO	MKP and MMPPFO	Updates pheromone based on resource consumption and fitness contribution	Enhances solution quality compared to standard ACO methods
[19]	FACO	Large TSP instances	Regulates additional edges in solutions; combines with other methods	Outperforms traditional ACO in finding high-quality solutions for large TSPs
[20]	ASPV	Optimal search path problem in rescue operations	Problem-specific pheromone initialization; iteration-best updates	Outperforms greedy heuristics and MILP solvers in larger instances
[21]	Free Ant and Restricted Ant	EMVRP	Clustering techniques to guide pheromone initialization	Free Ant achieves better solutions; Restricted Ant is faster for larger problems
[22]	Antovetalian	Network alignment	Adapts pheromone based on node similarity	Achieves near-optimal outcomes in network alignment tasks; strong tolerance to noise
[23]	SFACO	3D surface laser scanning path planning	Dynamic neighborhood structure and adaptive factors	Demonstrated efficacy in path planning; improved performance on complex surfaces
[24]	ACO-CPP	Edge traversal in linked undirected graphs	Creates shortest path matrices for pheromone updates	Effective and reliable for complex problems; better than traditional recursive and GAs

3 Mechanisms of ACO

An early version of the ACO algorithm, called AS, was utilized for comparatively fewer TSP cases of no more than 80 cities. Performance comparable to a few other widely used strategies, such as evolutionary computation [5, 6], was attained by the original AS, but it was insufficient to match or compete with the performance provided by other sophisticated algorithms created for TSP. Consequently, the ACO's research focus switched to ACO algorithms that perform superior to AS, including those utilized in the TSP.

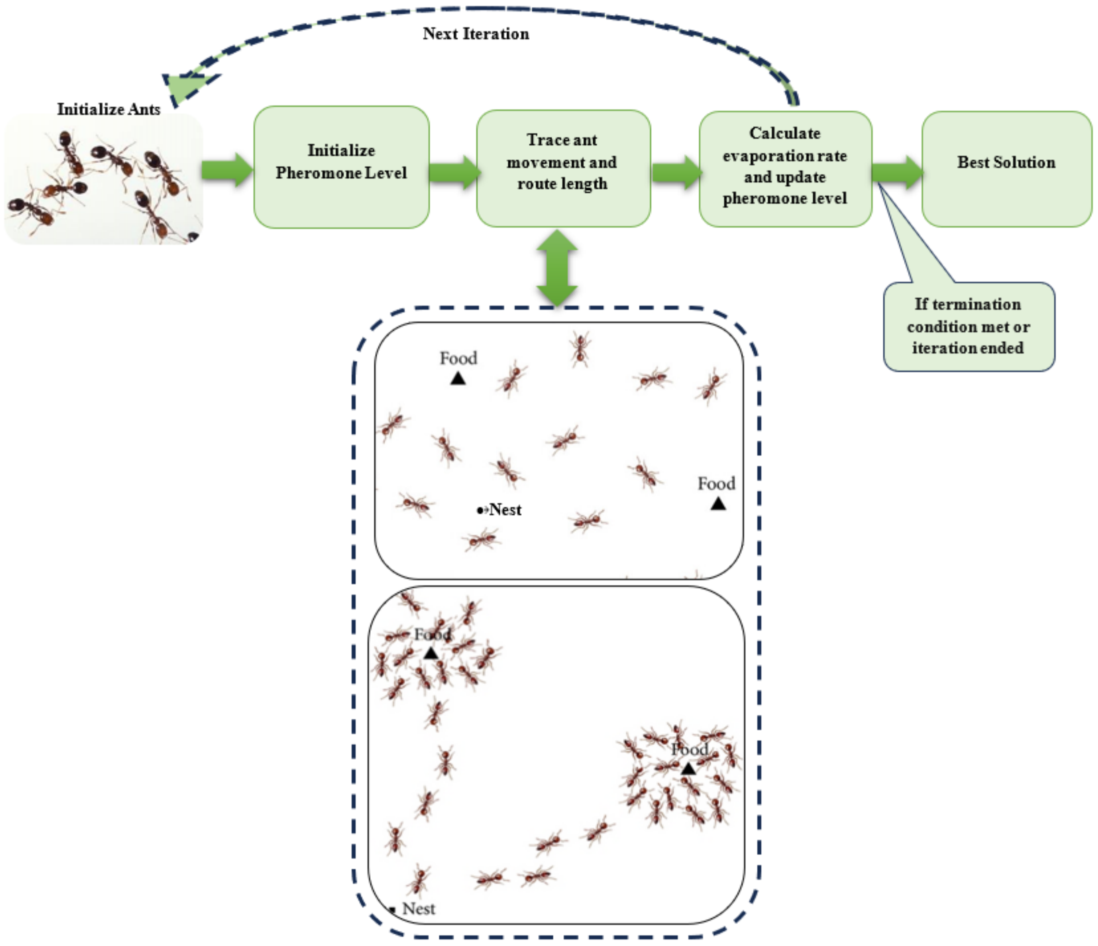
Next Iteration
Initialize Ants
Initialize Pheromone Level
Trace ant movement and route length
Calculate evaporation rate and update pheromone level
Best Solution
If termination condition met or iteration ended
Food
Nest
Food
Food
Food
Nest

Figure 1. Steps for ant colony optimization

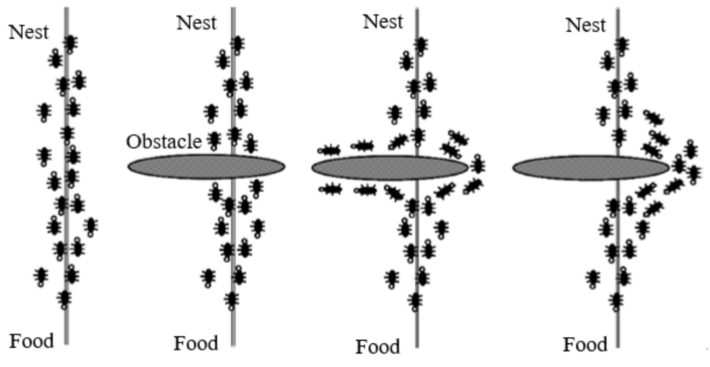

Figure 2. Inspiration from an ant colony looking for the best route between the food and the nest

The most distinctive feature of ant behavior is sociability. A stigmergy system exists wherever ants go within and around their ant colony because they use a chemical called pheromone to communicate to one another [25]. Pheromones are substances that ants, in many species, leave behind on the ground as they move from their colony to a food source. This pheromone is detectable by other ants, and its presence affects their path choice, causing them to gravitate toward areas with high pheromone concentrations. As shown in Figure 1, ants can locate good food supplies that have been previously identified by other ants by following the pheromone trail formed by the pheromone that is deposited on the ground [26]. According to the study by Deneubourg et al. [25], to better understand and investigate ant behavior related to pheromone laying and determining the shortest path to food, a controlled experimental setting was established. As shown in Figure 2, the ant colony and food supply were connected using a twin bridge of varying lengths for this reason. A shorter bridge and a longer bridge were used in many studies. The findings demonstrated that while ants first moved randomly in the direction of the food in the arena, they eventually adopted a shorter path. The explanation below applies to the results. Since no ants had explored the area at first, there was pheromone on the ground, which equalizes the likelihood of choosing any path without regard to preference. It may be concluded that, initially, approximately half of the ants chose each way. But compared to another group of ants that chose a longer path, the ones that chose a shorter path arrived at the food source and returned to the nest sooner. Ants now choose a path to take in order to return to their food source, and the shorter way has a higher pheromone level than the other, which influences their choice. Eventually, the pheromone levels on the shorter road begin to accumulate more quickly, leading almost the whole colony of ants to use that way. The same effect applies to many variants of ACO algorithms and ASs. Artificial ants (agents) are used in place of real ants. Pheromone trails are represented by artificial pheromone trails, and double bridge is shown by a graph. Artificial ants are equipped with additional ability to apply constraints and retrace solutions without errors in order to tackle real-world challenges. Artificial ants exhibit behavior akin to biological ants in that the quantity of pheromone they deposit is correlated with the quality of the solution they make [27].

3.1 Mathematical models of ACO

The concept behind ACO algorithms is to solve optimization problems by mimicking the cooperative behavior of real ants. ACO metaheuristics were first introduced by Dorigo [5]. They may be viewed as multiagent systems where the actions of individual agents are modeled after those of actual ants. ACO has historically been used to solve CO issues, and it has shown great effectiveness in doing so for a variety of issues, including scheduling, routing, and assignment [28].

An ant gives a tour of its solution. At first, every ant finds itself in a city at random. Next, using a probability-based procedure that takes into account the heuristic value and the amount of pheromone trails accumulated on the edge, each ant selects the next city. The random proportional rule is the name given to this formula. The number of cities in TSP (M, the number of ants, is set equal to N, the number of cities) is the number of TSP agents (artificial ants) used by the ACO algorithm. Given its location i, Eq. (1) provides the likelihood that ant k visits city j.

$$P_{i,j} = \frac{\tau_{i,j}^{\alpha} \cdot \eta_{i,j}^{\beta}}{\sum_{1 \in N_i^k} \tau_{i,j}^{\alpha} \cdot \eta_{i,j}^{\beta},}, \ j \in N_i^k \tag{1}$$

where, α is a parameter that controls the influence of $\tau_{i,j}$, and β is a parameter that controls the influence of $\eta_{i,j}$. Both α and β are factors that control the impact of pheromone trails and heuristic data and they have an impact on the next city selection process, as demonstrated below.

- If $\alpha < \beta$, it indicates that the closet city is more likely to be chosen.
- If $\alpha > \beta$, it indicates that the arcs that have more pheromone intensity are more likely to be used.
- If $\alpha = 0$, it indicates that the heuristic value is used for the selection without pheromone.
- If $\beta = 0$, it indicates that pheromone is used for the selection without the heuristic value, which may cause bad outcomes.

$\eta_{i,j}$ that gives in Eq. (2) is a priori available heuristic value; $distance(i, j)$ is the distance between cities i and j; $\tau_{i,j}$ is the pheromone trail matrix; and N_i^k is the set of the neighborhood that ant k has not visited yet. The probability that ant k chooses arc (i, j) increases with the pheromone trail value ($\tau_{i,j}$) and the heuristic value ($n_{i,j}$). The ACO, which is explained in Algorithm 1, consists of broad principles for creating foraging ant-based algorithms to address optimization issues.

$$\eta_{i,j} = \frac{1}{distance(i, j)} \tag{2}$$

Algorithm 1. ACO pseudo-code

```
Initialize parameters
Initialize pheromone trails
Create ants
while Stopping criteria are not met
    Let all ants construct their solution
    Update pheromone trails
    Allow daemon actions
end while
```

3.2 TSP

Testing and refining algorithms for the AS and algorithms based on ant behavior require a thorough understanding of the TSP [1]. The traveling salesman challenge served as the test subject for the 1992 AS debut of Dorigo [5]. TSP is an NP-hard CO problem [29]. If a group of cities is given, each city must be visited just once, and then the group must return to the starting city to complete the tour in the least amount of time feasible [30]. Symmetric TSP (STSP) and asymmetric TSP (ATSP) are the two main categories of TSP. The number of tours in the ATSP is $n-1$, but for n cities in the STSP, it is $(\ n-1)!/2$. Formally speaking, the TSP is a full weighted graph $G(N, A)$, where $A(i, j)$ is the collection of arcs linking the cities and N is the set of cities that must be visited. D_{ij} is a representation of the distance between the cities A_i and A_j. Therefore, as Eq. (3) below illustrates, the ideal (minimum length) journey to the TSP can be determined.

$$Btour = \left(\sum_{i=1}^{n-1} d_{p(i)p(i+1)} \right) + d_{p(n)p(1)} \tag{3}$$

where, p is a likelihood list of cities with minimum distance between the cities p_i and p_{i+1}.

4 Challenges Addressed by ACO

ACO algorithms are designed to tackle a variety of complex optimization problems. Below is a detailed explanation of the specific issues that ACO addresses, along with their complexity and significance.

4.1 CO Problems

ACO is particularly effective for CO problems, where the goal is to find the best arrangement or selection from a finite set of items. The TSP is a classic example, where the challenge lies in determining the shortest possible route that visits a set of cities and returns to the origin city. This problem is NP-hard, meaning that as the number of cities increases, the number of possible routes grows exponentially, making it computationally challenging to solve directly.

4.2 Dynamic and Uncertain Environments

ACO algorithms excel in dynamic environments where the optimization landscape can change over time. For instance, in routing problems, traffic conditions may vary, requiring the algorithm to adapt its solutions in real time. The ability of ACO to utilize pheromone trails allows it to adjust to these changes effectively, making it suitable for real-world applications like logistics and transportation. Even though ACO algorithms are strong instruments for resolving a variety of optimization issues, large levels of uncertainty and non-standard features can severely reduce their efficacy. It might be necessary to create hybrid strategies or alter the conventional ACO framework in order to overcome these constraints and improve its resilience and adaptability.

4.3 Multi-Objective Optimization

Many real-world problems involve multiple conflicting objectives that need to be optimized simultaneously. ACO can be adapted to handle multi-objective optimization, allowing it to find a set of optimal solutions that balance trade-offs between different objectives. This is significant in fields such as engineering design and resource allocation, where decisions must consider various factors.

4.4 Scalability and Efficiency

ACO algorithms are designed to be scalable, meaning they can handle large-size problems without a significant loss in performance. This is crucial for applications in industries like telecommunications and manufacturing, where the scale of optimization problems can be substantial. The efficiency of ACO in exploring large solution spaces while maintaining solution quality is a key advantage.

5 Current Applications of ACO

Metaheuristic algorithms undergo numerous modifications and improvements after their initial appearance to enable them to solve various problems [31]. ACO has been effectively applied to various applications in real-world scenarios, demonstrating its versatility and efficiency in solving complex optimization tasks. Following are some key applications:

5.1 Routing Problems

ACO is widely used in routing applications, such as vehicle routing and network routing. For instance, in logistics, ACO algorithms can optimize delivery routes for trucks, minimizing travel time and fuel consumption. The algorithm's ability to adapt to changing conditions, such as traffic or road closures, enhances its effectiveness in real-time routing scenarios [21].

5.2 Scheduling Tasks

In manufacturing and project management, ACO is employed to optimize scheduling tasks. It can efficiently allocate resources and schedule jobs to minimize completion time and maximize productivity. For example, ACO has been used to schedule tasks in production lines, ensuring that machines operate at optimal efficiency while meeting deadlines [11].

5.3 Assignment Problems

ACO algorithms are also effective in solving assignment problems, where tasks need to be assigned to resources in the most efficient manner. This includes applications in workforce management, where ACO can help assign employees to shifts based on their skills and availability, optimizing labor costs and improving service quality [28].

5.4 3D Surface Laser Scanning Path Planning

A notable application of ACO is in 3D surface laser scanning path planning. A study demonstrated the use of an adaptive ACO algorithm to determine the optimal scanning path for complex surfaces. By utilizing dynamic 3-opt neighborhood structures and adaptive heuristic factors, the algorithm effectively identified the best scanning route, significantly improving the efficiency of the scanning process [23]. This application highlights ACO's practical utility in fields requiring precision and efficiency, such as manufacturing and quality control.

5.5 CPP

ACO has been applied to solve the CPP, which involves finding the shortest path that traverses every edge in a graph. The ACO-CPP method effectively identifies odd nodes and creates a matrix of shortest pathways, demonstrating its capability in graph-based optimization problems [24]. This application is significant in urban planning and network design, where efficient routing is crucial.

6 Experimental Design in ACO Research

A thorough experimental framework must be established in order to improve the assessment of ACO algorithms. Performance measurements, benchmark issue selection, and a comparison of several ACO variations are all part of the system. The following are important factors to think about:

a) Benchmark problems

To guarantee varied application, a collection of well-known benchmark problems from many fields (such as the network routing problem, the vehicle routing problem, and the TSP) were chosen. These issues can make it possible to compare various ACO variations consistently.

b) Performance metrics

Each ACO variant's efficacy and efficiency were evaluated using established criteria, such as convergence speed, computation time, resilience, and solution quality (e.g., optimality and path cost). Depending on the application, measurements for resource usage and flexibility in changing contexts may also be added.

c) Comparative analysis

Several ACO algorithms were compared through experiments, such as hybrid algorithms, enhanced ACO versions (such as PCCACO and HACO-ABC-CHS), and standard ACO. The relative advantages and disadvantages of each version should be confirmed by statistical analysis of performance against benchmarks.

7 Comparison of ACO with Other Metaheuristic Approaches

To contextualize the strengths and limitations of ACO, this section provides a comparison with other metaheuristic approaches, such as GA and PSO.

a) Adaptability to dynamic environments

• ACO: It adapts solutions in real time by using pheromone trails, which makes it appropriate for dynamic contexts (such as network routing and logistics).

• GA/PSO: Although GA is able to preserve population diversity, PSO tends to converge more quickly but may not be as flexible if left alone.

b) Solution quality and convergence

• ACO: Because of pheromone reinforcement, it usually provides high-quality solutions. But if it is not adjusted properly, it may experience early convergence.

• GA: It employs crossover and mutation, which can preserve variety and prevent local optima, frequently leading to dependable convergence.

• PSO: It is known to converge quickly, particularly in simpler environments, although it can also converge too soon.

c) Computational efficiency

• ACO: It does well in combinatorial and discrete issues, but may need greater processing power for large-scale applications.

• GA: It strikes a balance between efficient exploration and computational requirements, particularly in high-dimensional search spaces.

• PSO: It benefits from straightforward updating equations and is typically computationally efficient.

d) Applications and suitability

• ACO: With its stigmergy-based search and optimization, it is ideally suited for routing, scheduling, and path planning.

• GA: It is adaptable to a variety of optimization contexts, such as design, scheduling, and machine learning applications.

• PSO: It is useful for real-time applications, including control systems and parameter adjustment, as well as continuous optimization.

8 Future Research Directions in ACO

The future of ACO research is promising, with numerous directions that can enhance its applicability and performance. By focusing on the directions below, the capabilities of ACO algorithms can be significantly advanced in solving complex optimization challenges.

a) Hybrid approaches

ACO could be combined with algorithms like deep learning and GA for better performance.

b) Real-time adaptation

ACO could be developed to work in dynamic and real-time environments, such as robotics and network routing.

c) Scalability

ACO's efficiency could be improved for large-scale applications through parallel computing and optimized resource use.

d) Enhanced exploration-exploitation

Adaptive strategies could be used to prevent premature convergence and improve solution quality.

e) New applications

ACO could be applied to emerging fields like renewable energy, climate modeling, and healthcare for broader impact.

9 Conclusion

There are several NP-hard issues in the family of CO problems. However, heuristics are better suited to address large instances of such issues since they typically use significantly fewer computer resources, even though certain small examples of such problems can be solved with exact approaches. ACO is a metaheuristic that was initially created to solve problems in the CO class, albeit its bounds have long ago been crossed. A subclass of swarm-based metaheuristic algorithms are AS techniques; the first algorithm utilizing ant behavior was created in 1992 by Dorigo [5]. Since its debut, numerous changes have been suggested for it, and it has proven capable of handling a wide range of issues in numerous domains. After providing an overview of the ant-based method, this study discusses some of its modifications. This study covers the applications from a variety of domains, such as engineering medicine, reliability optimization, power dispatch, and so on.

Data Availability

Not applicable.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] S. M. Almufti, "U-turning ant colony algorithm powered by great deluge algorithm for the solution of TSP problem," Master's thesis, Eastern Mediterranean University (EMU), 2015.

[2] P. Pinto, T. A. Runkler, and J. M. Sousa, "Wasp swarm optimization of logistic systems," in *Adaptive and Natural Computing Algorithms: Proceedings of the International Conference*, Coimbra, Portugal, 2005, pp. 264–267. https://doi.org/10.1007/3-211-27389-1_63

[3] M. Dorigo, G. Di Caro, and L. M. Gambardella, "Ant algorithms for discrete optimization," *Artif. Life*, vol. 5, no. 2, pp. 137–172, 1999. http://doi.org/10.1162/106454699568728

[4] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: An autocatalytic optimizing process," Department of Electronics, Politecnico di Milano, Milan, Italy, Tech. Rep., 1991.

[5] M. Dorigo, "Optimization, learning and natural algorithms," Ph.D. dissertation, Politecnico di Milano, Italy, 1992.

[6] C. Liu, L. Wu, W. Xiao, G. Li, D. Xu, J. Guo, and W. Li, "An improved heuristic mechanism ant colony optimization algorithm for solving path planning," *Knowl. Based Syst.*, vol. 271, p. 110540, 2023. http://dx.doi.org/10.1016/j.knosys.2023.110540

[7] G. Reinelt, *The Traveling Salesman: Computational Solutions for TSP Applications*. Springer Berlin, Heidelberg, 1994. https://doi.org/10.1007/3-540-48661-5

[8] M. Dorigo and L. M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 53–66, 1997. http://doi.org/10.1109/4235.585892

[9] R. R. Asaad and N. L. Abdulnabi, "Using local searches algorithms with Ant Colony Optimization for the solution of TSP problems," *Acad. J. Nawroz Univ.*, vol. 7, no. 3, pp. 1–6, 2018.

[10] S. Janakiraman, "A hybrid ant colony and artificial bee colony optimization algorithm-based cluster head selection for IoT," *Proc. Comput. Sci.*, vol. 143, pp. 360–366, 2018. http://doi.org/10.1016/j.procs.2018.10.407

[11] L. V. Tran, B. H. Huynh, and H. Akhtar, "Ant Colony Optimization algorithm for maintenance, repair and overhaul scheduling optimization in the context of industrie 4.0," *Appl. Sci.*, vol. 9, no. 22, p. 4815, 2019. https://doi.org/10.3390/app9224815

[12] H. W. Zhu, X. M. You, and S. Liu, "Multiple Ant Colony Optimization based on Pearson correlation coefficient," *IEEE Access*, vol. 7, pp. 61 628–61 638, 2019. http://doi.org/10.1109/ACCESS.2019.2915673

[13] H. J. Wang, F. Guo, H. F. Yao, S. S. He, and X. Xu, "Collision avoidance planning method of USV based on improved Ant Colony Optimization algorithm," *IEEE Access*, vol. 7, pp. 52 964–52 975, 2019. http://doi.org/10.1109/ACCESS.2019.2907783

[14] W. Gao, "New Ant Colony Optimization algorithm for the traveling salesman problem," *Int. J. Comput. Intell. Syst.*, vol. 13, no. 1, pp. 44–55, 2020. http://doi.org/10.2991/ijcis.d.200117.001

[15] Y. Liu and B. Cao, "A novel Ant Colony Optimization algorithm with levy flight," *IEEE Access*, vol. 8, pp. 67 205–67 213, 2020. http://doi.org/10.1109/ACCESS.2020.2985498

[16] M. C. Wang, C. Y. Zhu, F. Y. Wang, T. T. Li, and X. Y. Zhang, "Multi-factor of path planning based on an Ant Colony Optimization algorithm," *Ann. GIS*, vol. 26, no. 2, pp. 101–112, 2020. http://doi.org/10.1080/19475683.2020.1755725

[17] S. Liang, T. Jiao, W. Du, and S. Qu, "An improved Ant Colony Optimization algorithm based on context for tourism route planning," *PLOS ONE*, vol. 16, no. 9, p. e0257317, 2021. http://doi.org/10.1371/journal.pone.0257317

[18] J. Skackauskas, T. Kalganova, I. Dear, and M. Janakiram, "Dynamic impact for Ant Colony Optimization algorithm," *Swarm Evol. Comput.*, vol. 69, p. 100993, 2022. http://doi.org/10.1016/j.swevo.2021.100993

[19] R. Skinderowicz, "Improving Ant Colony Optimization efficiency for solving large TSP instances," *Appl. Soft Comput.*, vol. 120, p. 108653, 2022. http://doi.org/10.1016/j.asoc.2022.108653

[20] M. Morin, I. Abi-Zeid, and C. G. Quimper, "Ant Colony Optimization for path planning in search and rescue operations," *Eur. J. Oper. Res.*, vol. 305, no. 1, pp. 53–63, 2023. http://doi.org/10.1016/j.ejor.2022.06.019

[21] N. Frías, F. Johnson, and C. Valle, "Hybrid algorithms for energy minimizing vehicle routing problem: Integrating clusterization and Ant Colony Optimization," *IEEE Access*, vol. 11, pp. 125 800–125 821, 2023. http://doi.org/10.1109/ACCESS.2023.3325787

[22] G. R. Corominas, M. J. Blesa, and C. Blum, "Antnetalign: Ant Colony Optimization for network alignment," *Appl. Soft Comput.*, vol. 132, p. 109832, 2023. https://doi.org/10.1016/j.asoc.2022.109832

[23] J. F. Song, Y. Y. Pu, and X. Y. Xu, "Adaptive Ant Colony Optimization with sub-population and fuzzy logic for 3D laser scanning path planning," *Sensors*, vol. 24, no. 4, p. 1098, 2024. http://doi.org/10.3390/s24041098

[24] G. A. Sgarro and L. Grilli, "Ant Colony Optimization for Chinese postman problem," *Neural Comput. Appl.*, vol. 36, pp. 2901–2920, 2024. http://doi.org/10.1007/s00521-023-09195-4

[25] J. L. Deneubourg, S. Aron, S. Goss, and J. M. Pasteels, "The self-organizing exploratory pattern of the Argentine ant," *J. Insect Behav.*, vol. 3, pp. 159–168, 1990. http://doi.org/10.1007/BF01417909

[26] S. Katiyar, I. Nasiruddin, and A. Q. Ansari, "Ant Colony Optimization: A tutorial review," in *National Conference on Advances in Power and Control*, Faridabad, India, 2015, pp. 99–110.

[27] R. Beckers, J. L. Deneubourg, and S. Goss, "Modulation of trail laying in the ant lasius niger (Hymenoptera: Formicidae) and its role in the collective selection of a food source," *J. Insect Behav.*, vol. 6, pp. 751–759, 1993. https://doi.org/10.1007/BF01201674

[28] M. Dorigo and T. Stützle, "The Ant Colony Optimization metaheuristic: Algorithms, applications and advances," in *Handbook of Metaheuristics*. Kluwer Academic Publishers, 2002, pp. 251–285. https://doi.org/10.1007/0-306-48056-5_9

[29] S. M. Almufti, R. B. Marqas, P. S. Othman, and A. B. Sallow, "Single-based and population-based metaheuristics

for solving NP-hard problems," *Iraqi J. Sci.*, vol. 62, no. 5, pp. 1710–1720, 2021. https://doi.org/10.24996/10.24996/ijs.2021.62.5.34

[30] C. Blum, "Ant Colony Optimization: Introduction and recent trends," *Phys. Life Rev.*, vol. 2, no. 4, pp. 353–373, 2005. http://doi.org/10.1016/j.plrev.2005.10.001

[31] S. M. Almufti, "Historical survey on metaheuristics algorithms," *Int. J. Sci. World*, vol. 7, no. 1, pp. 1–12, 2019. http://doi.org/10.14419/ijsw.v7i1.29497