Acta Informatica Pragensia
2026, Volume 15, Issue 3, 719–739
Special Issue: Selected Extended Papers from International Conferences CITA, ICRAMI, and ADACIS
https://doi.org/10.18267/j.aip.326

PRAGUE UNIVERSITY OF ECONOMICS AND BUSINESS

Review Open Access

Coordination in Multi-Agent Systems: Overview of Metaheuristic Approaches

Ferial Laassami, Mohammed Elhabib Souidi, Abdeldjalil Ledmi

Knowledge Engineering and Computer Security Laboratory, Department of Computer Science, Abbes Laghrour Khenchela University, Khenchela, Algeria

Corresponding author: Ferial Laassami (ferial.laassami@univ-khenchela.dz)

Editorial Record

First submission received:
February 2, 2026

Revisions received:
February 24, 2026
June 13, 2026
July 26, 2026

Accepted for publication:
August 5, 2026

Special Issue Editors:
Waad Alhoshan
Imam Mohammad Ibn Saud Islamic University, Saudi Arabia

Subrata Chakraborty
University of New England, Australia

Hakim Bendjenna
Echahid Cheikh Larbi Tebessi University, Algeria

Academic Editor:
Zdenek Smutny
Prague University of Economics and Business, Czech Republic

This article was accepted for publication by the Academic Editor upon evaluation of the reviewers' comments.

How to cite this article:
Laassami, F., Souidi, M. E., & Ledmi, A. (2026). Coordination in Multi-Agent Systems: Overview of Metaheuristic Approaches. *Acta Informatica Pragensia*, 15(3), 719–739. https://doi.org/10.18267/j.aip.326

Abstract

Background: In the field of distributed artificial intelligence, multi-agent systems (MAS) collaborate to resolve complex problems. To operate efficiently, these systems require effective coordination mechanisms. Metaheuristic algorithms are now standard for decentralized optimization and agent task distribution.

Objective: This study evaluates the strengths and limitations of common metaheuristics used for MAS path planning and task coordination. Additionally, it introduces comparative simulation studies to evaluate the performance of baseline and dynamic metaheuristic methods specifically in MAS path planning.

Methods: This paper presents a comprehensive narrative review of metaheuristic approaches for MAS coordination to identify current research gaps. It evaluates recent literature to identify their strengths, weaknesses and future trends. For practical validation, this review is coupled with a comparative simulation study using NetLogo to evaluate 50 agents in a 2D grid. The study specifically compares the standard crow search algorithm (SCSA) and standard particle swarm optimization (PSO) against their adaptive and modified versions (AAP-CSA, DEC-DAPCSA, MCSA, BPSO and LDPSO). This dual approach effectively bridges the gap between theoretical overview and practical performance evaluation.

Results: The findings indicate that metaheuristic algorithms significantly affect MAS performance. Specifically, standard algorithms (SCSA and PSO) often trap in local optima, but their dynamic variants improve both fitness and convergence speed.

Conclusion: The evaluated dynamic parameter strategies outperformed standard versions in both solution quality and search time. Ultimately, selecting the right metaheuristic and applying dynamic parameter control are essential for solving complex MAS coordination problems.

Index Terms

Multi-agent control; Metaheuristics; Task coordination; Path planning; Particle swarm optimization; Crow search algorithm.

1 INTRODUCTION

Metaheuristics (Abdel-Basset et al., 2018) represent advanced computational frameworks inspired by diverse domains, including biology, physics, mathematics and artificial intelligence, to guide other heuristics towards better solutions. By adaptively balancing global exploration (searching new regions) and local exploitation (refining known solutions), these algorithms efficiently navigate high-dimensional search spaces to identify optimal solutions and escape local optima.

In parallel, multi-agent systems (MAS) emerged as a distinct field of study around 1980 and gained widespread recognition in the mid-1990s (Wooldridge, 2012). They consist of multiple specialized agents used in artificial intelligence, robotics, traffic management, industrial processes, military operations, electronic markets and manufacturing. According to McArthur et al. (2007), an agent is an autonomous system that senses its environment and acts upon it, responding dynamically to changes. While single-agent systems struggle with large-scale, complex problems, MAS distribute responsibilities across multiple collaborating entities. Consequently, effective coordination is critical for both path planning (Sariff & Buniyamin, 2006) and task allocation (Yazdanpanah et al., 2020). Path planning enables agents to identify optimal routes, avoid obstacles and reduce conflicts, while task coordination optimizes how they manage responsibilities.

Path planning and task coordination in MAS are addressed using both centralized and decentralized techniques (Mas & Kitts, 2010; Sharma et al., 2016). To solve them, the current scientific landscape is broadly categorized into several methodological paradigms:

- Classical and deterministic algorithms: Traditional graph search methods such as A* and multi-label A* (MLA) (Grenouilleau et al., 2019) or geometric bug algorithms (Souidi et al., 2017) provide exact path guarantees. However, their computational complexity grows exponentially as the number of agents and spatial constraints increase.
- Learning-based and game-theoretic frameworks: Approaches such as reinforcement learning (e.g., IMAP-QL for pursuit-evasion games) (Souidi et al., 2024), game theory for strategic interactions (Jin et al., 2025) and digital twin approaches for multi-objective optimization (Sahu et al., 2025) handle dynamic multi-agent interactions well, but often require heavy computational overhead.
- Metaheuristic frameworks: They bridge the gap between computational scalability and solution quality. Broadly categorized into single-agent metaheuristics algorithms (SMAs) and population-based metaheuristic algorithms (PMAs) (Houssein et al., 2024), these techniques utilize concurrent search mechanics that naturally mirror the decentralized nature of MAS.

Specifically, population-based metaheuristic algorithms (PMAs) are exceptionally well-suited for MAS due to their inherent structural alignment with distributed execution. PMAs are broadly divided into two major families. The first one includes evolutionary algorithms (EAs), such as genetic algorithms (GA) (El-Shorbagy, 2026), which rely on selection, crossover and mutation operators to evolve candidate paths over successive generations. Secondly, there are swarm intelligence (SI) techniques, such as particle swarm optimization (PSO) (Jia et al., 2021) and the crow search algorithm (CSA) (Meraihi et al., 2021), utilize cooperative velocity update rules and position shifts inspired by biological swarms.

By processing multiple candidate solutions simultaneously, PMAs enable decentralized control across large agent groups (Król & Drożdżowski, 2010). Their parallel search mechanics explore complex spatial landscapes efficiently while maintaining scalability (Milano & Roli, 2004). Swarm variants further optimize agent positions using low-overhead communication to balance individual and collective efforts (Souidi et al., 2023).

However, baseline PMAs suffer from severe structural vulnerabilities when applied to dynamic MAS navigation, specifically:

- parameter sensitivity: algorithm efficacy depends heavily on static, manually tuned hyperparameters;
- premature convergence and local optima trapping: when static control parameters favour localized exploitation too early, the swarm loses its global exploration capability, consequently leading to suboptimal solutions.

While the narrative review provides a broader taxonomy of MAS navigation and planning strategies, the experimental section focuses specifically on evaluating swarm-based metaheuristics (PSO and CSA variants) as representative solutions to parameter sensitivity and local convergence. These two families were specifically selected because they directly exemplify the structural vulnerabilities identified in the literature while offering adaptable, low-overhead position update mechanisms well-suited for distributed 2D grid environments.

Building upon the identified challenges, this manuscript investigates metaheuristic optimization for MAS path planning and task coordination within dynamic environments. To bridge existing research gaps, the present study is driven by the following primary objectives:

- synthesizing and comparing various metaheuristic approaches applied to MAS path planning and task coordination to clearly identify current structural limitations;
- conducting a comprehensive simulation study to evaluate and compare the performance of the CSA and PSO and their advanced dynamic parameter extensions in solving MAS path planning challenges;
- validating the efficacy of dynamic parameter control in resolving the specific operational limitations observed in standard baseline algorithms; and
- extracting key findings and providing recommendations to guide the future selection and deployment of metaheuristics in complex, dynamic MAS environments.

This paper is associated with and was presented at the International Conference on Recent Advances in Mathematics and Informatics (ICRAMI 2025), Tunisia. The authors have expanded the work (Laassami et al., 2025) by providing additional simulation results by incorporating parameters from another algorithm.

The structure of this paper is as follows: Section 1 offers a brief introduction to the research context. It also includes a structured review of recent literature concerning MAS path planning and task coordination. Section 2 details the methodology, beginning with an overview of the fundamental metaheuristic algorithms used in the study, followed by the setup for a comparative case study utilizing the CSA and PSO. Section 3 presents the empirical results, featuring the outcomes of both case studies. Section 4 makes a detailed discussion synthesizing experimental findings with theoretical literature. Section 5 concludes the paper by summarizing the key findings and results of the analysis.

1.1 Literature selection

To clearly understand the current literature, a review was conducted to map and connect the most relevant metaheuristic approaches used in MAS coordination. The literature search relied primarily on Google Scholar, focusing on papers indexed within major academic databases such as Scopus, Web of Science and IEEE Xplore. Research published between 2016 and 2025 was prioritized using the search string: ("multi-agent system" OR "MAS") AND ("path planning" OR "task coordination") AND ("metaheuristic"). Studies that explicitly focus on decentralized agent architectures for path planning or task coordination were selected. Each study was analysed to identify advantages, limitations and future directions, with the final summary presented here to highlight recent progress.

To ensure clarity regarding the mechanisms of the metaheuristic frameworks, Table 1 provides a brief overview of the biological or physical inspirations and primary search mechanisms of the eight baseline algorithms discussed in this review. Section 2 details the governing equations and mathematical breakdown of these frameworks.

Table 1. *Foundational principles of baseline metaheuristics.*

Algorithm	Core inspiration	Search mechanism
Particle swarm optimization (PSO)	Social foraging behaviour of bird flocks	Updates agent trajectories using both personal (*pBest*) and global (*gBest*) positions.
Ant colony optimization (ACO)	Collective foraging strategies and trail-tracking mechanisms of ants	Employs indirect pheromone-based agent communication.
Artificial bee colony (ABC)	Collective honeybee foraging behaviour	Divides the population into employed, onlooker and scout roles to optimize both global and local search spaces.
Genetic algorithms (GA)	Evolutionary genetics	Drives population evolution across successive generations.
Firefly algorithm (FA)	Flashing light behaviour of fireflies	Directs agents towards brighter neighbours.
Cuckoo search algorithm (CS)	Cuckoo brood parasitism mechanisms	Uses parasitic egg-laying behaviour combined with scale-free Lévy flights for global exploration.
Whale optimization algorithm (WOA)	Cooperative bubble-net feeding mechanisms	Employs mathematical encircling and spiral updates for prey capture.
Crow search algorithm (CSA)	Intelligent food-caching behaviour of crows	Uses spatial memory and awareness probability for position updates.

1.1.1 Application in path planning

Yahia & Mohammed (2023) evaluated metaheuristic algorithms for MAS path planning focusing on performance across dynamic environments, robotic navigation and autonomous vehicle transportation. They provided a comprehensive analysis of metaheuristic applications in unmanned aerial vehicle (UAV) path planning, detailing their operational advantages, technical limitations and future developmental perspectives. Moreover, Dhulkefl and Durdu (2019) provided an extensive evaluation of diverse path planning algorithms specifically engineered for UAVs in various operational scenarios. They further investigates Saeed et al. (2022), who focused on employing swarm intelligence techniques to elevate the path planning performance of drones.

Accordingly, works were selected as samples for analysis, which are detailed in the result section, including Biswas et al. (2017), Nayeem et al. (2021) and Sun & Niu (2024), which revolve around applying the PSO method (see Section 2.1.1 for foundational principles) to solve path planning challenges within MAS. The first proposed a PSO-based framework for cooperative path planning and task assignment. This approach focuses on improving how agents coordinate within the system. The second one introduced a modified PSO algorithm specifically for UAVs. This method improves path planning by adjusting the inertia weight (w) parameter, which makes the core PSO perform better than the first approach. The third approach addressed traditional PSO limitations by providing a variety of search locations and adjusting behaviour based on the problem. The method also uses cubic spline interpolation for smoother paths. Lastly, Najm et al. (2024) introduced a hybrid approach that combines the WOA and PSO. This integration is designed to make path planning more efficient and reliable.

In addition, Trujillo-Romero (2022) compared the ABC and ACO (detailed mathematically in Section 2.1) metaheuristic algorithms for mobile robot path planning. The results show that the ABC algorithm is superior when used in specific environments. Moreover, there are other works such as Huang et al. (2021) and Zheng et al. (2018) that address multi-agent path planning (MAPP) challenges by using an enhanced ACO algorithm. Both works utilize this improved method to solve complex MAS path planning problems. Besides this Bahabadi et al. (2024) employed a MAS group that uses a hybrid version of the ACO and GA (see Section 2.1.4 for core principles) algorithms. This strategy allows the agents to cover specific areas more efficiently. Regarding the ABC algorithm, it reviews many works that demonstrate the flexibility of this approach for MAS path planning. These studies show how well the method handles challenges in dynamic environments, for example Majumder et al. (2016).

Turning to the GA, numerous studies have explored solutions to MAPP, such as Lamini et al. (2018), which featured a novel crossover operator for GA-based path planning within static environments. This specific operator improves how the GA combines potential solutions to find the best path. Moreover, Rath et al. (2021) designed a hybrid controller for humanoid robot path planning by combining neural networks and the GA. Lastly, a novel hybridization was presented by Sood & Panchal (2024). Their approach combines the FA and the CS algorithms (formulated mathematically in Section 2.1) for MAS path planning.

Table 2 provides an analysis of these metaheuristic algorithms for MAS path planning, summarizing their strengths, limitations and future research directions as identified in the literature.

Table 2. *Analysis of metaheuristics algorithms for MAS path planning.*

Metaheuristics	Reference	Strength	Limitation	Future perspectives
PSO	Biswas et al. (2017)	- enhancing the safety of paths (treating other agents as dynamic obstacles) - encourages cooperation - adaptability for various scenarios (different simulations with various environmental complexities)	- limited scalability (number of agents)	- applying the framework to a larger number of agents
	Nayeem et al. (2021)	- similar set of advantages to that of Biswas et al. (2017) - balancing exploration / exploitation - addressing the limitation of trapping in local optima (it is not mentioned in Biswas et al., 2017)	- parameter sensitivity (w) - limited scalability (offline path planning is less scalable for dynamic environments)	- multivariable w

Metaheuristics	Reference	Strength	Limitation	Future perspectives
	Sun & Niu (2024)	- preventing local optima in complex 3D environments - effective path planning for multi-AUVs (autonomous underwater vehicle) with high security, while Nayeem et al. (2021) focused on the PSO algorithm itself	- algorithm handles highly unpredicted obstacles and changes in the environment in real-time	- enhancing real-time coordination - incorporating real-time sensing to address unpredictable environments
	Najm et al. (2024)	- integrating PSO's rapid convergence with WOA's deep exploration capabilities - balancing exploration / exploitation	- complexity of implementation - limited scalability (lacks evidence for large-scale robot performance) - parameter sensitivity	- real and dynamic environments testing - exploration of other hybridizations
ACO	Trujillo-Romero (2022)	- ensuring collision-free paths	- parameter sensitivity (performance is highly dependent on chosen values)	- creating an algorithm that is less sensitive to parameter tuning
	Huang et al. (2021), Zheng et al. (2018)	- handling static and dynamic OBSTACLES - using an improved ACO avoids local optima	- limited scalability (computational costs scale with the number of agents and the size of the environment)	- real-time applications - applying it to more complex scenarios
	Bahabadi et al. (2024)	- agents cooperate even though they have different characteristics	- algorithms might perform badly in contexts that change fast (offline or static planning approach)	- hybridizing strategies for environmental changes in real-time
ABC	Trujillo-Romero (2022)	- ensuring collision-free paths - the ABC algorithm outperformed ACO	- parameter sensitivity (performance is highly dependent on chosen values)	- creating an algorithm that is less sensitive to parameter tuning
	Majumder et al. (2016)	- flexibility and convergence speeds due to its stochastic nature	- limited scalability (high computational costs for managing pheromones and agent interactions as system size and complexity increase) -trapping in local optima	- hybridizing strategies for real-time scenarios
GA	Lamini et al. (2018)	- creating feasible paths	- algorithms might perform badly in a dynamic environment	- applying the algorithm to dynamic environments
	Rath et al. (2021)	- close alignment of simulation and experiment	- difficulty of combining GA/ neural network (NN), with sensitive parameter selection	- collaborative control among multiple humanoid robots
FA and CS	Sood & Panchal (2024)	- capitalizing on the exploration of the FA and the exploitation of CS	- parameter sensitivity - computational complexity	- hybridization with parameter tuning

1.1.2 Application in task coordination

Research is currently exploring the potential of metaheuristics to optimize complex task allocation and scheduling within MAS coordination. This study opens with an overview of Biswas et al. (2017), which presented a PSO framework for cooperative task assignment. This is followed by the discrete differential evolution PSO (DEPSO) hybrid approach, which integrates DE and PSO to handle discrete task allocation in epidemic scenarios (Ma et al., 2022).

It presents another approach by Long et al. (2023) that optimizes scheduling and task distribution for multiple satellites. This method combines the GA with simulated annealing (SA) to improve performance. Furthermore, Y. Wang et al. (2021) suggested the max-min ant system as an enhanced ACO that uses graph theory. S. Wang et al. (2022), introduced the multi-objective ant colony system (MOACS) method, which is designed to distribute tasks to robots efficiently.

Priya & Sahana (2022) discussed a hybrid algorithm using the CSA (formulated mathematically in Section 2.1) for task scheduling in multiprocessor systems. Finally, Evaznia & Ebrahimi (2023) presented a multi-objective CSA for optimizing resource management.

Table 3 analyses these metaheuristic algorithms for MAS task coordination, summarizing their strengths, limitations and future research directions as identified in the literature.

Table 3. *Analysis of metaheuristics algorithms for MAS task coordination.*

Metaheuristics	Reference	Strength	Limitation	Future perspectives
PSO	Biswas et al. (2017)	- coordinating tasks among several agents - encourages cooperation (agents negotiate task assignments)	- scalability issues (the total computation time increases with the increase in task assignment) - agent heterogeneity (in terms of differing capabilities or roles)	- developing a more efficient task assignment strategy
	Ma et al. (2022)	- relevance to real-world scenarios - balancing exploration / exploitation	- limited scalability (in large-scale scenarios with many tasks)	- dealing with complex scenarios
GA	Long et al. (2023)	- managing a large number of satellites and tasks	- complex implementation combining GA and simulated annealing (SA) - parameter sensitivity (specific parameters for task scenarios)	- optimizing energy usage for satellites and hybridizing
ACO	Y. Wang et al. (2021)	- performance in terms of completion time	- the algorithms might perform badly in dynamic environments (frequent changes, unexpected events)	- hybridization of the enhanced ACO with other techniques
	S. Wang et al. (2022)	- reducing both travel distance and total task completion time	- complexity of computing time (challenges scalability to massive multi-robot systems)	- handling dynamic and online scenarios
	Priya & Sahana (2022)	- encouraging exploration and exploitation	- complexity of computing time and computational resources	- real-time scheduling scenarios
CSA	Evaznia & Ebrahimi (2023)	- balancing resource utilization and energy efficiency	- limited scalability (the centralized approach limits its scalability for massive, real-world cloud data centres) - computational complexity	- hybridizing and investigating scalability

1.1.3 Literature research gaps and motivation

The synthesized literature highlights the inherent limitations and advantages that show how critical it is to select the appropriate metaheuristic for both path planning and task coordination challenges. This selection cannot be arbitrary; it must account for architectural factors such as scalability, environmental complexity, computational efficiency and the dynamic balance between exploration and exploitation. It identifies recurring limitations in these existing perspectives, such as parameter sensitivity and local optima trapping (as synthesized in Tables 2 and 3). To clearly map these structural vulnerabilities, Table 4 establishes a concise taxonomy that categorizes some of the reviewed metaheuristics according to their primary operational limitations.

Table 4. *Taxonomy of metaheuristics categorized by primary operational limitations.*

Primary operational limitation	Corresponding metaheuristics
Parameter sensitivity	CSA GA ACO PSO
Local optima trapping (premature convergence)	PSO ACO
Exploration vs exploitation imbalance	ABC (strong exploration / weak exploitation) ACO (strong exploitation/ weak exploration) GA (strong exploration / weak exploitation)
Scalability & computational complexity	GA ACO

Accordingly, this paper bridges the gap between narrative literature synthesis and simulation validation. It employs multiple distinct dynamic parameter strategies across two algorithmic families (AAP-CSA, DEC-DAPCSA, MCSA, BPSO and LDPSO, which will be detailed in Section 2.2) within a unified NetLogo simulation environment. This framework allows us to explicitly demonstrate how directly addressing parameter sensitivity prevents stagnation at local optima in decentralized agent behaviour, focusing on their convergence speed and ability to achieve optimal fitness levels in MAS environments. Concurrently, based on these simulated behavioural insights and the literature synthesis, the study provides secondary recommendations for deploying alternative metaheuristics within specific MAS solutions.

While the present work resolves parameter tuning in standard architectures, it underscores that future MAS research must focus on structured hybrid MAS algorithms to maximize solution quality and maintain resilience when operating within highly complex, large-scale and unpredictable dynamic environments.

2 OVERVIEW OF ALGORITHMS AND METHODOLOGY

This section is divided into two main parts: first, it describes the core principles and mathematical equations of the most popular and widely applied metaheuristics; then, it details the technical design and parameterization of the comparative simulation frameworks.

2.1 Overview of metaheuristic algorithms

To provide a comprehensive evaluation of MAS coordination, eight metaheuristic algorithms were selected based on their prevalence in recent literature and their diverse approaches to solving both path planning and task coordination. This subsection details the fundamental mechanisms underlying each selected metaheuristic.

2.1.1 Particle swarm optimization

PSO is a well-known metaheuristic algorithm (Eberhart & Kennedy, 1995) that mimics the social foraging behaviour of bird flocks, where individual particles adjust their trajectories towards their own best-known positions and the global best-found solution. It utilizes particles as MAS agents that interact with one another and their environment to collectively converge on optimal solutions. The position of a particle in the search space is evaluated via an objective function, while its velocity governs the rate of displacement; these positions are iteratively updated according to Equation (1) (Gupta & Srivastava, 2020):

$$x\,(t\,+\,1)\,=\,x(t)\,+\,v(t\,+\,1) \tag{1}$$

Where x is the position and v is the velocity at a given iteration t.

The velocity is calculated using Equation (2) as follows (Gupta & Srivastava, 2020):

$$v(t + 1) = wv(t) + c1r1\big(pBest(t) - x(t)\big) + c2r2\,(gBest(t) - x(t)) \quad (2)$$

Where *pBest* represents the optimal solution identified by an individual, *gBest* represents the global best solution, *c1, c2* are the acceleration coefficients, *r1, r2* are random parameters from the range [0,1], *w* is the inertia weight, *pBest* is updated through a comparison with the new position of the particle at each iteration.; The *gBest* position is used to describe the position that had the highest fitness value, while *w* controls the exploration and exploitation.

2.1.2 Ant colony optimization

ACO (Dorigo, 1992) is a biologically inspired metaheuristic based on the foraging behaviour of ants, widely applied to dynamic challenges such as the travelling salesman problem (TSP) and the quadratic assignment problem (QAP) (Guntsch & Middendorf, 2002). The algorithm utilizes artificial ants to iteratively construct solutions through pheromone-based communication, where increasing pheromone concentrations on high-quality paths guide the swarm towards optimal solutions. The probability of selecting the component $p(c_i \mid s)$ is calculated using Equation (3) (Wong & Komarudin, 2008):

$$p(ci \mid s) = \frac{[\tau_i]^{\alpha}.[\eta(c_i)]^{\beta}}{\sum_{c_j \in N(s)}[\tau_i]^{\alpha}.[\eta(c_i)]^{\beta}} \quad (3)$$

Where s represents the current partial solution, c_i represents a solution component, τ_i refers to the pheromone value of c_i, and $\eta(c_j)$ denotes the heuristic desirability associated with the feasible solution component $c_j \in N(s)$. The coefficients α and β are positive parameters that control influence of the stored memory (pheromone) and the locally available information (heuristic).

The pheromone update mechanism encourages exploration of better solutions. It is updated using Equation (4) in the following way (Wong & Komarudin, 2008):

$$\tau i \leftarrow (1 - \rho).\tau i + \rho. \sum_{\{s \in S_{upd} \mid c_i \in s\}} w_s F(s) \quad (4)$$

Where S_{upd} denotes a collection of solutions selected, $\rho \in [0, 1]$ is the evaporation rate, τi refers to the trail intensity that persists post-evaporation, $F : S \rightarrow R+$ is the quality function given that $f(s) < f(s') \rightarrow F(s) \geq F(s')$, $\forall s \neq s' \in S$, w denotes a weight applied to a solution s to modulate the quality function, with $f(s)$ referring to the objective function. The evaporation rate controls pheromone decay speed.

2.1.3 Artificial bee colony

ABC (Karaboga, 2005) draws from honeybee conduct, partitioning the population into three types: employed, onlooker and scout. Employed bees explore local neighbourhoods, onlooker bees select candidate solutions based on quality and spatial metrics and scout bees execute random searches to identify novel food sources. Bees function as agents that iteratively acquire and share knowledge regarding the distribution and quality of food sources within the search space. Also, an onlooker bee selects a potential food source based on its associated probability value p_i calculated using Equation (5) (Gao et al., 2013):

$$p_i = \frac{f\,it_i}{\sum_{j=1}^{S\,N} f\,it_j} \quad (5)$$

Where $i = 1,2..., SN$. SN is the size of employed bees or onlooker bees, $j = 1,2..., D$. D represents the problem dimension, $f\,it_i$ is the fitness value of solution i. Each solution is initialized using Equation (6) (Gao et al., 2013):

$$x_{i,j} = x_{min,j} + rand(0, 1)(x_{max,j} - x_{min,j}) \quad (6)$$

Where $x_{min,j}$, $x_{max,j}$ define the variable bounds. These apply to the dimension j. For generating a new solution V_i, the algorithm takes an existing solution X_i, selects another solution X_k and the difference between X_i and X_k is weighted by a random factor φij in [-1, 1] and then added to X_i. Where $k \in \{1,2...,SN\}$ and $j \in \{1,2...,D\}$, Equation (7) shows that. If a solution stagnates, it is replaced using Equation (6).

$$v_{i,j} = x_{i,j} + \varphi_{i,j}(x_{i,j} - x_{k,j}) \quad (7)$$

2.1.4 Genetic algorithm

The GA (Sampson, 1976) utilized natural selection to generate solutions using binary strings, with the best individuals selected for the next generation. The crossover and mutation operators facilitate the exchange of genetic material and the introduction of stochastic variations, respectively, to enhance population diversity and improve solution quality. The new generation replaces the lowest-fitness individuals. The mapping of MAS agents to GA individuals facilitates the execution of evolutionary operations through simulated social interactions and information exchange. The roulette wheel selection technique, represented by Equation (8), provides a stochastic mechanism for choosing individuals based on their proportional fitness within the population (Lipowski & Lipowska, 2012):

$$P(xi) = \frac{f(xi)}{\sum_{j=1}^{N} f(xj)} \tag{8}$$

Where N is the total population size and $f(xi)$ is the fitness of the individual xi.

2.1.5 Firefly algorithm

The FA (Bazi et al., 2021) is a recent metaheuristic. It involves random initialization of a population within the search space, where agents are assigned fitness values via an objective function and iteratively update their positions towards brighter individuals to balance exploration and mutual attractiveness. The attractiveness variation β based on the distance r (here r is equal to 0) and β_0 (denotes the attractiveness) can be expressed in Equation (9) (Kumar & Kumar, 2021):

$$\beta = \beta_0 e^{-\gamma r^2} \tag{9}$$

Let us consider two fireflies, denoted as x_i and x_j. The movement of the first one in the direction of the second one is shown in Equation (10) (Kumar & Kumar, 2021):

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{ij}^2} (x_j^t - x_i^t) + \alpha_t \epsilon_t \tag{10}$$

Where $\beta_0 e^{-\gamma r_{ij}^2}$ represents the attractiveness of the j-th firefly to the i-th firefly, $(x_j^t - x_i^t)$ denotes the distance vector between the two fireflies, $\alpha_t \epsilon_t$ represents a stochastic component to the firefly's movement, ϵ_t is a random vector with the same dimension as the search space. It generates a random direction in the search space. The parameter α_t scales this random direction, determining the step size.

2.1.6 Cuckoo search algorithm

The CS (Yang & Deb, 2010) is based on the behaviour of cuckoo birds. It utilizes the concept of brood parasitism, where new candidate solutions replace existing ones based on performance; poor solutions are abandoned in favour of global exploration facilitated by Lévy flights. This method is beneficial for complex optimization tasks (Akbari & Rashidi, 2016). Equation (11) is used to update a solution $Xi^{(t+1)}$ using Lévy flight (Joshi et al., 2017):

$$Xi^{(t+1)} = Xi^{(t)} + \alpha \oplus Lévy(\lambda) \tag{11}$$

Where $Xi^{(t)}$ is the current position of the cuckoo i, α denotes the scaling factor, $\oplus$ is element-wise multiplication, $Lévy(\lambda)$ represents a Lévy flight random variable with the parameter λ.

2.1.7 Whale optimization algorithm

The WOA (Mirjalili & Lewis, 2016) mimics the hunting strategies of humpback whales. It uses the bubble-net feeding strategy of humpback whales, initializing a search population to evaluate fitness and dynamically calculate the spatial distance between individual agents and the target prey. The algorithm selects a stochastic number p in the range [0, 1] to manage the search. The encircling mechanism enables agents to iteratively update their positions relative to the prey, a process that continues until the predefined termination criteria are satisfied. If p is less than 0.5, the process of encircling the prey satisfies Equation (12) (Gharehchopogh & Gholizadeh, 2019). It provides the distance D of an agent $X(t)$ from the current best solution, $X^*(t)$. The vector C is the coefficient that governs the wrapping behaviour.

$$\vec{D} = |\vec{C} \cdot \vec{X}^*(t) - \vec{X}(t)| \tag{12}$$

The vector C is calculated using Equation (13) (Gharehchopogh & Gholizadeh, 2019), where $\vec{r}$ represents a random vector in [0, 1]. Then, the updating position process is acquired with Equation (14) (Gharehchopogh & Gholizadeh, 2019):

$$\vec{C} = 2 . \vec{r} \tag{13}$$

$$\vec{X}(t+1) = \vec{X}^*(t) - \vec{A} . \vec{D} \tag{14}$$

Where A is the coefficient with which the algorithm operates among the exploration vs exploitation process. The agents move away from the best solution when $|A| > 1$, exploring new areas. While in the exploitation phase when $|A| < 1$, they refine the current solution. It is calculated using Equation (15), where $\vec{a}$ decays linearly from 2 down to 0 (Gharehchopogh & Gholizadeh, 2019). In the second case, the spiral position updating is achieved via Equation (16) (Gharehchopogh & Gholizadeh, 2019):

$$\vec{A} = 2 \vec{a} . \vec{r} - \vec{a} \tag{15}$$

$$\vec{X}(t+1) = \vec{D}' . e^{bl} . cos(2\pi l) + \vec{X}^*(t) \tag{16}$$

Where $\vec{D}' = |\vec{X}^*(t) - \vec{X}(t)|$ and l represents a stochastic value in [-1,1]. The constant b is used to determine the logarithmic form of the spiral.

2.1.8 Crow search algorithm

The CSA is a metaheuristic inspired by the intelligent foraging behaviour of crows, specifically their ability to store food in hidden caches and retrieve it through spatial memory. The simplicity, efficiency and robustness of the algorithm facilitate its broad application to complex optimization challenges across engineering, computer science and various multidisciplinary domains (Hussien et al., 2020). The mathematical modelling of CSA formalizes the social dynamics of crows, prioritizing the role of spatial memory to maintain a strategic equilibrium between global exploration and local exploitation. It initializes a population, randomly selecting crows to steal food, and adjusts position based on awareness probability. The crow updates its position using Equation (17) (Hussien et al., 2020):

$$x_{i,iter+1} = \begin{cases} x_{i,iter} + r_i \times fl_{i,iter} \times (m_{j,iter} - x_{i,iter}) & r_j \geq AP_{j,iter} \\ A\ random\ position & otherwise \end{cases} \tag{17}$$

Where $x_{i,iter}$ represents the current position of the crow i, r_i, r_j denote random numbers, $AP_{j,iter}$ is the awareness probability of the crow j, $fl_{i,iter}$ represents the flight length of the crow i to remember the crow j. If the new position $x_{i,iter+1}$ has a better fitness value than the current memory $m_{i,iter}$, the memory is updated using Equation (18) (Hussien et al., 2020):

$$m_{i,iter+1} = \begin{cases} x_{i,iter+1} & f(x_{i,iter+1}) \leq f(m_{i,iter}) \\ mi, iter & otherwise \end{cases} \tag{18}$$

2.2 Case study setup

To provide practical insights that extend beyond our theoretical overview, the case study directly addresses the primary gaps identified in the literature review, namely local optima trapping and parameter sensitivity. Consequently, this case study serves as a targeted validation, designed to show how dynamic parameter control can mitigate the specific issues identified.

Accordingly, this case study serves as a targeted validation, designed to show how dynamic parameter control can mitigate the specific issues identified. A case study is first proposed, utilizing standard CSA (SCSA) (Hussien et al., 2020) and other extended versions, known as adaptive awareness probability-based CSA (AAP-CSA) (Mandala & Chandra Sekhara Rao, 2019), decreasing dynamic awareness probability CSA (DEC-DAPCSA) (Necira et al., 2022) and modified CSA (MCSA) (Islam et al., 2019), allowing MAS path planning in a partially observable environment.

The choice of CSA over all others is due to its computational efficiency and robustness. It relies on minimal parameters, which makes it easier to use than algorithms requiring a dozen variables. Specifically, the CSA primarily uses only two parameters: *fl* and *AP*. These parameters are crucial for controlling the balance between exploration and exploitation. The *AP* determines whether a crow moves towards a random location or approaches a known food source. Consequently, a high *AP* pushes the algorithm towards exploration. Complementing this, the *fl* regulates the

magnitude of the displacement: higher values facilitate global leaps into unexplored regions, whereas lower values permit precise local refinement of the current solution.

Beside this, the case study is also based on the use of the PSO (Gupta & Srivastava, 2020) and other extended versions known as Butterworth PSO (BPSO) (Zhu & Wang, 2018) and linearly decreasing PSO (LDPSO) (Xin et al., 2009).

The choice of PSO over all others is due to its computational efficiency and robustness. This study examines the impact of the parameter w. It can be represented by a constant value or adjusted dynamically during the execution process. This parameter controls the balance between exploration and exploitation of the search space. On the one hand, a high w favours exploration. On the other hand, a low w favours exploitation. Usually, the w values used are in the range of 0.4 to 0.9.

Additionally, these algorithms are suitable for dynamic MAS, showcasing their ability to model agent interactions and collective intelligence.

To conduct the simulations for this study, we employed the NetLogo platform (Tisue & Wilensky, 2004), version 5.3.1. NetLogo plays a crucial role in achieving robust simulation results, particularly in areas such as agent-based modelling. Furthermore, this case study involves 50 agents interacting in a 2D grid of 100×100 environmental positions, while the simulation environment is represented with each possible position having a specific fitness degree between 0 and 1. Specifically, these fitness values are stochastically (randomly) generated across the grid upon initialization. The target position is assigned a fitness value of 1, identifying it as the unique global optimum.

The fitness function $F(xi)$ is mathematically equivalent to the objective function. Consequently, an agent's fitness is determined directly as $F(xi) = Value(xi)$, where xi represents the agent's current coordinates and $Value(xi)$ denotes the static, predefined fitness score assigned to that specific location. Although this landscape serves as an abstract benchmark environment without explicit physical obstacles or collision penalties, the MAS coordination challenge lies in collectively searching this landscape without getting trapped in local sub-optimal positions. Consequently, maximizing $F(xi)$ in minimal algorithm iterations directly maps search performance to standard MAS metrics: guiding agents along direct spatial trajectories minimizes individual step counts (path length/flowtime), while collective rapid convergence minimizes overall scenario completion time (makespan).

To implement a visual approximation of the fitness degree of patches, we employed the native colour palette of the NetLogo platform. An increase in fitness degree was represented by a corresponding increase in patch brightness.

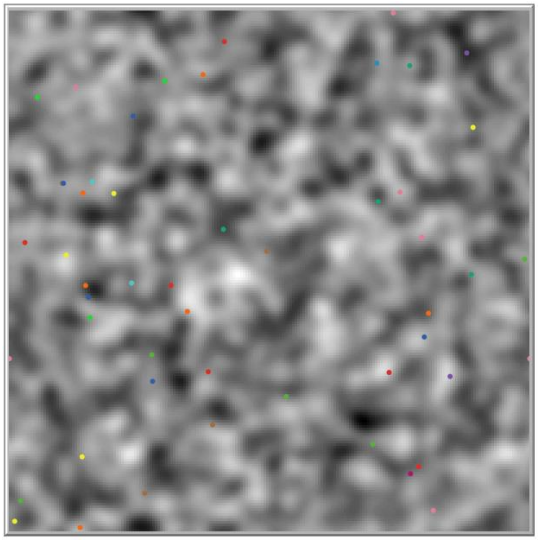

Figure 1. *Initial state of the environment.*

In the first case, each agent acts as a crow and updates its position (represented by Cartesian coordinates) iteratively according to Equations (17) and (18). Figure 1 illustrates the initial random distribution of agents across the simulation environment. The fitness levels of the various patches are dynamically updated between simulation episodes to reflect changes in the environment, noting that each agent is represented with a circle with a random

colour. The first simulation in Figure 2 involved episodes of 50 iterations. It focuses on the *AP* and *fl* development during the execution of the four compared cases. They are explained as follows:

- SCSA (Hussien et al., 2020): this case study utilizes the original CSA version detailed in Section 2. In this version, *AP* generally remains 0.1 while *fl* generally remains 0.5 during the algorithm iterations.
- AAP-CSA (Mandala & Chandra Sekhara Rao, 2019): in this case, the *AP* is changing during the different iterations using Equation 19 (decreases to encourage exploration then exploitation).

$$APj = 1 - (iter/itermax) \quad (19)$$

Where *iter* represents the current iteration and *itermax* refers to the maximum number of iterations.

- DEC-DAPCSA (Necira et al., 2022): for decreasing dynamic awareness probability, it will be decreased linearly from 0.1 to 0.9 over iterations using Equation (20).

$$APiter = \left(\frac{APmin - Apmax}{Iter-max}\right) \times iter + APmax \quad (20)$$

Where *Iter-max* refers to the maximum number of iterations and *iter* refers to the current iteration.

- MCSA (Islam et al., 2019): in this case, *fl* and *AP* are adjusted using Equations (21) and (22). The *AP* value starts from 0 and increases to 1. The *fl* value starts from 2.5 and decreases.

$$fl = 2*\frac{tmax-titr}{tmax} + 0.5 \quad (21)$$

$$AP = 1 - \frac{tmax-titr}{tmax} \quad (22)$$

Here, *tmax* denotes the maximum iteration number and *titr* is the current iteration number. At the initial iteration, *fl* and *AP* start with 2.5 and 0, respectively.

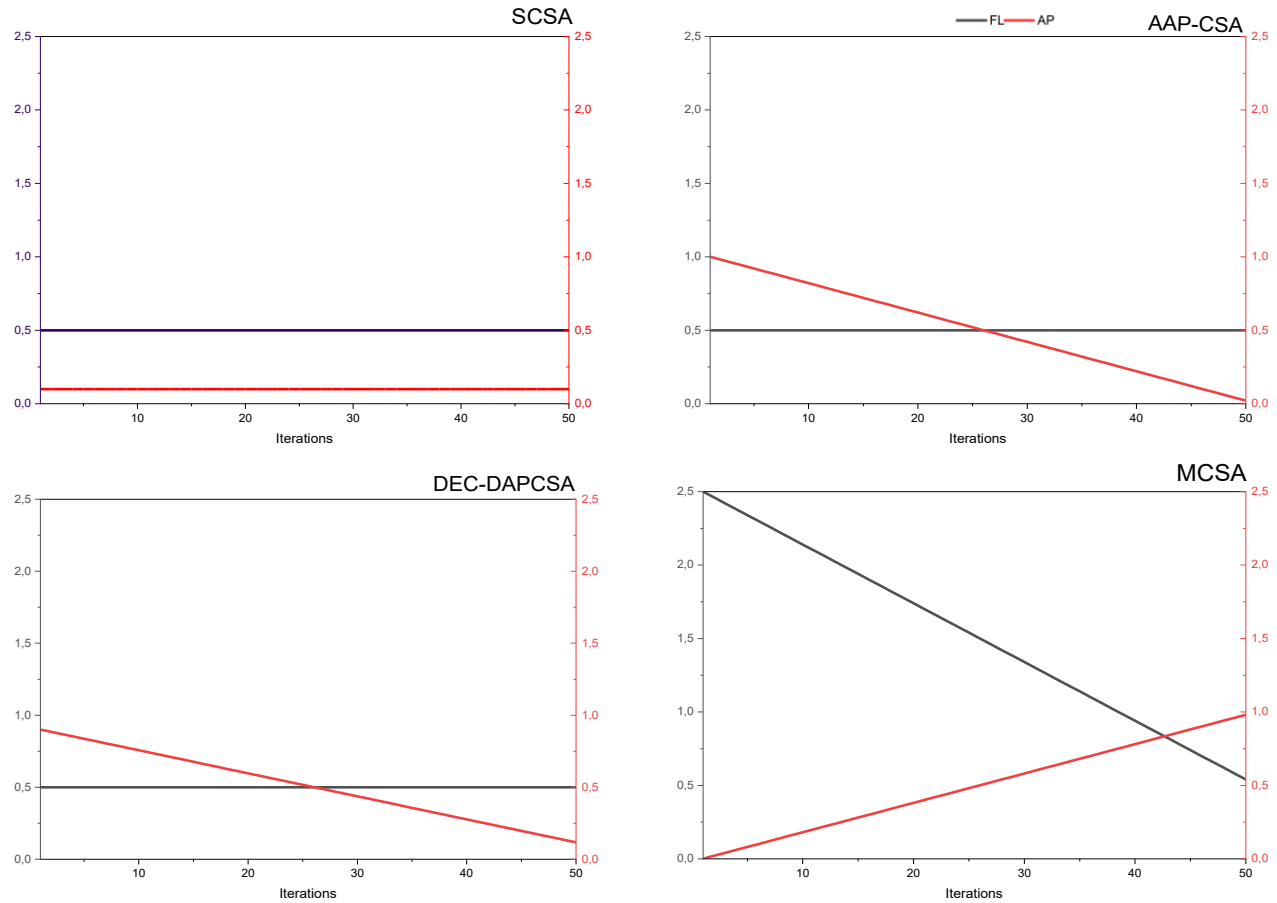

Figure 2. *Development of AP and fl during an episode of 50 iterations in 4 CSA cases.*

In the second case, each agent acts as a particle and updates its position iteratively according to Equations (1) and (2). As with the first case, the simulation baseline and fitness degree mechanisms remain unchanged. The environmental layout, visual colour scales and dynamic patch updates are identical to the setup described in the CSA cases. The constant parameters *c1* and *c2* are fixed at the beginning of each simulation episode. However, the parameters *r1* and *r2* are randomly updated before each iteration. The first simulation in Figure 2 involved episodes of 50 iterations. It focuses on the w development during the execution of the three compared cases. They are explained as follows:

- PSO (Gupta & Srivastava, 2020): this case is based on the original PSO version explained in section 2.1.1 of this paper. In this version, w is equal to 1 during the algorithm iterations.
- BPSO (Zhu & Wang, 2018): in this PSO version, the authors proposed a non-linear decrease of the w parameter as shown in Figure 3 using Equation (23) (Zhu & Wang, 2018):

$$w_i = w_{max} * \frac{1}{1+(\frac{t}{p1})^{p2}} + w_{min} \quad (23)$$

  Where i is the current iteration, $p1 = \frac{MaxIter}{3}$, $p2 = 10$, $w_{max} = 0.5$, $w_{min} = 0.4$.

- LDPSO (Xin et al., 2009): in this PSO version, the authors proposed a linear decrease of the w parameter as shown in Figure 3 using Equation (24) (Xin et al., 2009):

$$w_i = w_{min} - \frac{Wmin - Wmax}{MaxIter} * i \quad (24)$$

  Where $w_{min} = 0.4$, $w_{max} = 0.9$.

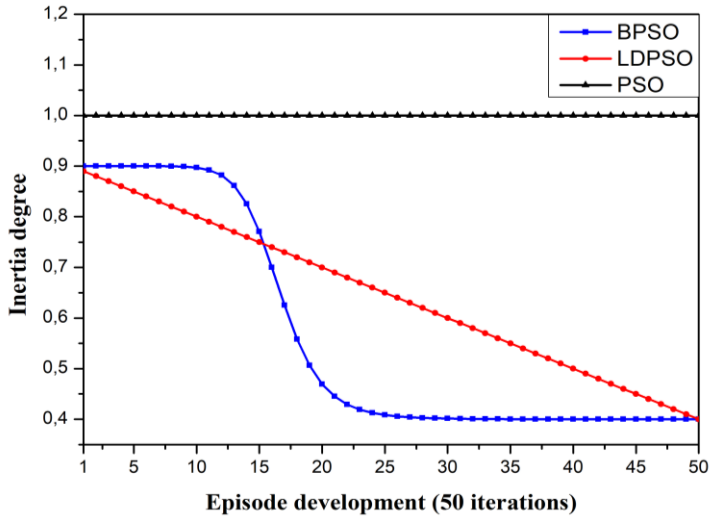

Figure 3. *Inertia weight development during 50 iterations in 3 PSO cases.*

3 RESULTS

This section presents the empirical results from the simulation study designed in Subsection 2.2, highlighting the performance of the adaptive CSA and PSO variants.

3.1 CSA cases

Figure 4 represents the maximum fitness degree obtained by the crows during 50 episodes in the four compared cases. Each episode is composed of 100 iterations. The SCSA case saw agents achieve the maximum fitness degree in 1/50 episodes, AAP-CSA in 5/50 episodes, DEC-DAPCSA in 13/50 episodes and MCSA in 16/50 episodes. The average fitness degree in SCSA during 50 episodes is (0.884), with an increase in AAP-CSA (0.97), DEC-DAPCSA (0.978) and MCSA (0.98) cases. According to Table 5, the three enhanced variants achieve target localization rates (fitness = 1) of 32% for MCSA, 26% for DEC-DAPCSA and 10% for AAP-CSA. Conversely, the standard SCSA algorithm yields the lowest performance, resulting in an average fitness degree of 0.884 and a target localization rate of only 2%.

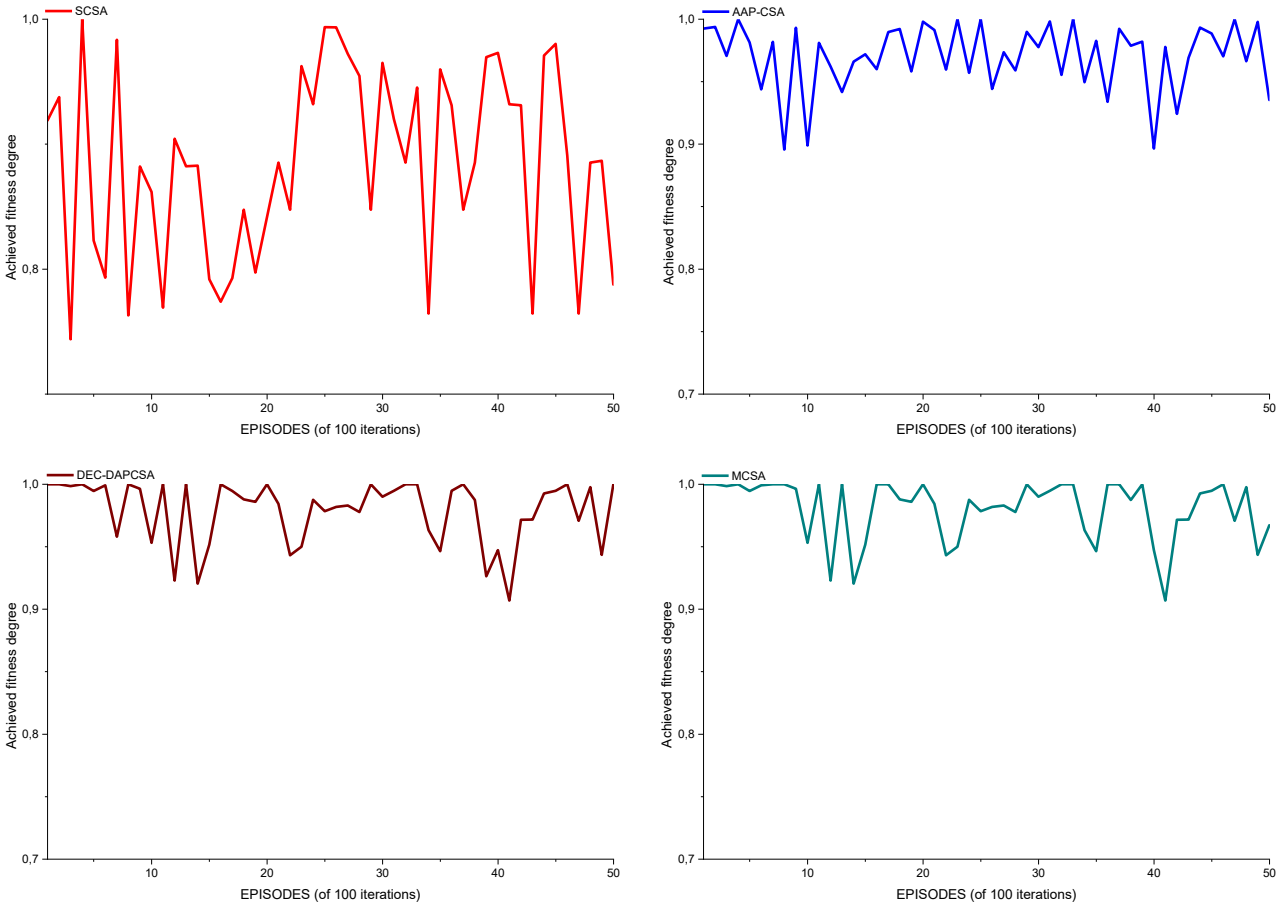

Figure 4. *The achieved fitness degree during 50 episodes (of 100 iterations) in 4 CSA cases.*

Table 5. *Results of CSA variants.*

	SCSA	AAP-CSA	DEC-DAPCSA	MCSA
Average fitness degree obtained in 50 episodes	0.884	0.97	0.978	0.98
Average of cases where (fitness degree = 1) in 50 episodes	2%	10%	26%	32%

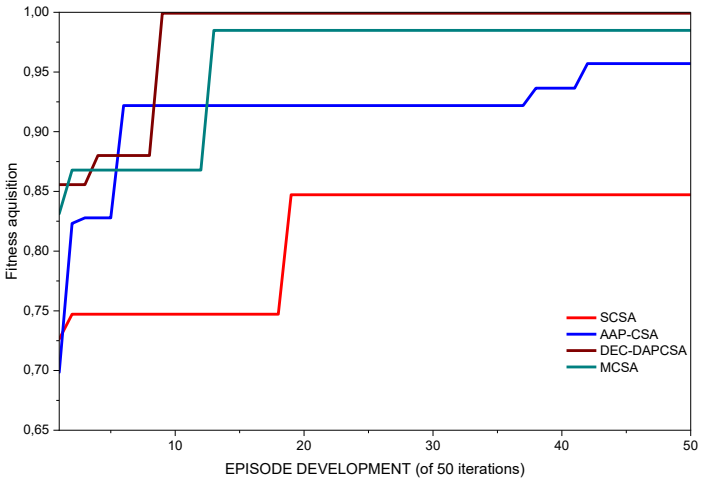

Figure 5. *Fitness development during complete episode (of 50 iterations) in 4 CSA cases.*

Figure 5 reflects the development of fitness (the highest fitness obtained in each iteration by the crow) during one complete episode of 50 iterations. The study reveals that the AAP-CSA case achieves the maximum fitness degree

(0.957) after 43 iterations, while DEC-DAPCSA achieves it (0.999) after only 10 iterations and the MCSA case achieves 0.984 after 14 iterations. The SCSA version gives 0.847 after 20 iterations.

3.2 PSO cases

Figure 6 represents the maximum fitness degree obtained by the swarm during 100 episodes in the three compared cases. Each episode is composed of 50 iterations. The PSO case saw agents achieve maximum fitness degree in 3/100 episodes, BPSO in 25/100 episodes and LDPSO in 46/100 episodes. The average fitness degree in PSO during 100 episodes is (0.912), with an increase in BPSO (0.931) and LDPSO (0.956) cases. According to Table 6, the evaluated variants reflect target localization rates (fitness = 1) of 46% for LDPSO and 25% for BPSO over 100 episodes. Conversely, the standard PSO algorithm yields the lowest average fitness degree of 0.912 and a target localization rate of only 3%.

Figure 7 reflects the development of fitness (the highest fitness obtained in each iteration by the swarm) during one complete episode of 50 iterations. The study reveals that the LDPSO case achieves the maximum fitness degree (0.959) after only 10 iterations, while the BPSO case achieves it (0.935) after 21 iterations. The PSO version achieves 0.911 after 35 iterations. It suffers from stagnation within a local maximum solution with a fitness degree of 0.873 in early iterations.

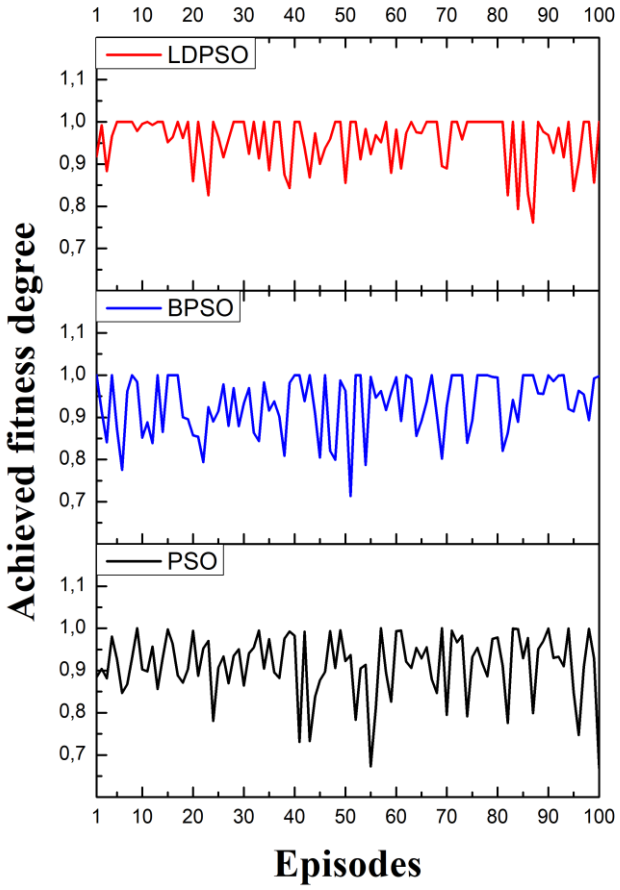

Figure 6. *Achieved fitness degree during 100 episodes in 3 PSO cases.*

Table 6. *Results of PSO variants.*

	PSO	BPSO	LDPSO
Average fitness degree obtained in 100 episodes	0.912	0.931	0.956
Average of cases where (fitness degree = 1) in 100 episodes	3%	25%	46%

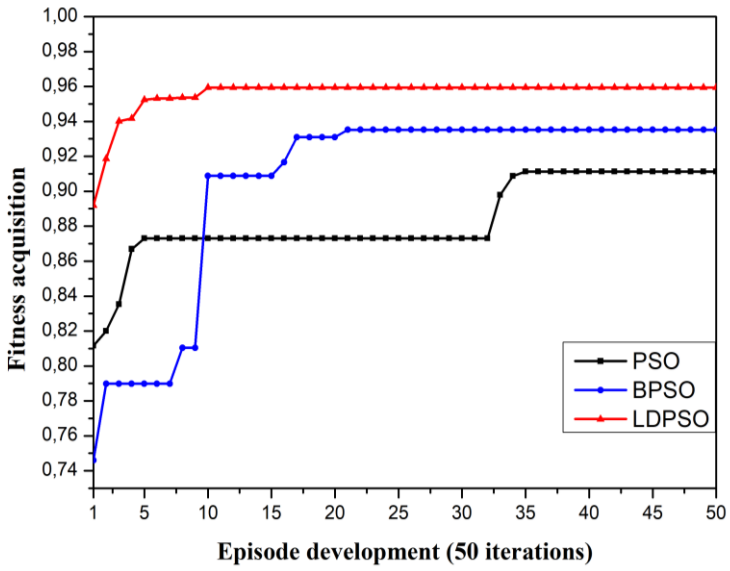

Figure 7. *Fitness development during complete episode (of 50 iterations) in 3 PSO cases.*

4 DISCUSSION

The NetLogo simulations reinforce and extend the trends previously identified. Specifically, the following observations were made.

4.1 Empirical validation from NetLogo experiments

For CSA variants:

- The fact that the SCSA got stuck at local minima of 0.847 (as shown in Figure 4) provides direct proof of the local optima trapping and the parameter sensitivity limitations noted in Table 2.
- The rapid convergence and high fitness acquisition demonstrated by DEC-DAPCSA (achieving a maximum fitness of 0.999 after only 10 iterations) and MCSA (achieving 0.984 after 14 iterations) confirm the future research directions noted in Table 2, which are parameter tuning to govern the balance between exploration and exploitation.
- Increasing the AP value encourages initial global exploration of the search space, while adjusting the fl ensures a smooth transition towards the best solutions found.

For PSO variants:

- The stagnation of standard PSO at a local maximum of 0.873 in the early iterations (Figure 7) directly validates the local optima trapping limitations noted in our literature review. This structural vulnerability restricts the standard algorithm to a low target localization rate of only 3% and the lowest average fitness degree of 0.912 over 100 episodes (Table 6).
- The rapid fitness acquisition of the advanced variants highlights the importance of adaptive parameter control. As illustrated in Figure 7, LDPSO achieves a high fitness degree of 0.959 after only 10 iterations, outperforming BPSO (0.935 after 21 iterations) and standard PSO (0.911 after 35 iterations). This acceleration enables LDPSO to successfully locate the absolute target in 46% of the 100 episodes compared to BPSO (25%) and PSO (3%), as confirmed in Figure 6.
- Dynamic control of the parameter w provides strong goal orientation and eliminates premature convergence. LDPSO encourages broad global exploration during early iterations, while a lower w later ensures a smooth transition towards exploitation.

4.2 Synthesis of empirical findings with theoretical literature

Consequently, synthesizing these empirical findings with the literature leads to several critical validations:

- Hybrid frameworks such as crow swarm optimization (CSO) combine the global exploration strengths of CSA with localized exploitation capabilities (Laassami et al., 2026). Integrating these segregated metaheuristic mechanisms and velocity adjustment rules substantially mitigates MAS collision risks while optimizing path trajectory accuracy.
- PSO variants reflect modern trends in parameter-driven MAS path planning. Specifically, they mirror the architectural strategies addressed in the inversed Butterworth framework (IB-PSO) (Laassami et al., 2024). The system relies on dynamic regulation of the w. This control governs the transition between exploration and exploitation. Consequently, the adaptive strategy successfully prevents the swarm from getting trapped in local optima.
- This study bridges the gap between the theoretical reviews found in current literature and practical algorithmic execution, offering a concrete roadmap for parameter optimization in real-world MAS coordination problems.

From a theoretical perspective, these results highlight that standard metaheuristics struggle to adapt when navigating complex, decentralized spaces. Their efficiency does not just come from adding more agents to the MAS. It depends on how smoothly those agents can shift from exploring the environment to exploiting the best paths. By introducing dynamic parameter tuning, this study offers a straightforward mathematical way to build smarter agent behaviour without adding heavy computational overhead. On a practical level, these findings give engineers a clear, real-world guide for deploying physical entities such as drones or warehouse robots. Using frameworks such as DEC-DAPCSA, MCSA or LDPSO allows these robots to identify optimal paths much faster, which directly optimizes battery life and keeps navigation responsive even when environmental coordinates change. The rapid convergence of these adaptive strategies proves that agents can recalculate optimal trajectories when facing an unpredictable, changing environment.

4.3 Guidelines for metaheuristic selection in MAS coordination

Additionally, based on our results from both the literature synthesis and the simulation study, several simple guidelines can be offered for selecting metaheuristics for MAS coordination. Our recommendations are categorized into two classes as follows.

Empirically validated guidelines:

- For low-power physical systems running on hardware with limited computing capacity, the CSA framework is highly recommended, as it relies primarily on only two main parameters (*AP* and *fl)*, achieving rapid convergence with minimal computational overhead. Meraihi et al. (2021) validated this perspective.
- CSA variants may be deployed to mitigate premature stagnation.
- For cooperative path planning and obstacle avoidance, PSO is ideal for coordinating agent positions safely due to its superior local exploitation and fine-tuning capabilities. This finding aligns with the results reported by Jia et al. (2021).

Literature-derived guidelines:

- For real-time or dynamic environments, ABC should be selected because it possesses high exploration capabilities (Moradi et al., 2018), which are essential for closing the resilience gap during shifting coordination conditions.
- ACO demonstrates unique suitability for heterogeneous MAS coordination (Loisel, 2023) by enabling effective cooperation across agents with diverse operational characteristics. This is achieved through a decentralized mechanism in which agents communicate indirectly by modifying their shared environment.
- GA (El-Shorbagy, 2026) offers high resistance to local optima stagnation by constantly exploring new regions of the search space. It relies entirely on the fitness scores generated by the simulation, allowing extreme flexibility in optimizing complex agent behaviour.
- For complex, multi-objective problems requiring a balance between conflicting goals such as distance and time, hybrid frameworks should be chosen (e.g., CSO (Laassami et al., 2026)) rather than standalone methods to ensure the necessary algorithmic flexibility.

5 CONCLUSION

In this study, we reviewed the literature regarding application of metaheuristics in path planning and task coordination in MAS. Key concepts were introduced and cases were analysed to understand their advantages and limitations. Some cases, such as PSO, GA and ACO, were evaluated in path planning, while others in task coordination. The study found common structural vulnerabilities across these cases, specifically regarding parameter sensitivity and local optima trapping. To demonstrate the use of metaheuristics for MAS coordination, especially in path planning and to empirically validate these theoretical insights, we proposed comparative case studies that used several versions of the CSA and PSO algorithms.

The study demonstrated that the adaptive AAP-CSA, DEC-DAPCSA, MCSA, BPSO and LDPSO strategies enhance solution quality, reduce search time and address local maximum issues. This emphasizes the significance of dynamic control of parameters (such as AP, fl and w). To maintain methodological rigour and bridge the literature synthesis with our experimental validation, our final guidelines explicitly distinguish between findings experimentally proven through these NetLogo simulations (PSO and CSA variants) and broader theoretical directions derived from the literature (ACO, GA, ABC). Ultimately, the study underscores how overcoming parameter sensitivity is crucial for system efficiency. Future work should aim to develop other enhanced versions of MAS algorithms or hybridize them by combining strengths and weaknesses, addressing complex environments, adjusting parameters and improving adaptability and convergence speed for real-world optimization problems.

ADDITIONAL INFORMATION AND DECLARATIONS

Acknowledgments: The authors would like to thank the reviewers for their valuable comments and suggestions.

Conflict of Interests: The authors declare no conflict of interest.

Author Contributions: F.L., M.E.S., and A.L.: Conceptualization, Methodology, Software, Validation, Formal analysis, Writing – Original draft preparation, Writing – Reviewing and Editing, Visualization. The authors contributed equally to this work.

Statement on the Use of Artificial Intelligence Tools: The authors declare that artificial intelligence tools were used for language editing and grammar refinement of the manuscript. No AI tools were used for generating scientific ideas, analysis, or data interpretation.

Data Availability: The data that support the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

Abdel-Basset, M., Abdel-Fatah, L., & Sangaiah, A. K. (2018). Metaheuristic Algorithms: A Comprehensive Review. In *Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications* (pp. 185–231). Elsevier. https://doi.org/10.1016/B978-0-12-813314-9.00010-4

Akbari, M., & Rashidi, H. (2016). A multi-objectives scheduling algorithm based on cuckoo optimization for task allocation problem at compile time in heterogeneous systems. *Expert Systems with Applications*, 60, 234–248. https://doi.org/10.1016/j.eswa.2016.05.014

Bahabadi, M. H. J., Mahdavi, A., & Khankalantary, S. (2024). Heterogeneous Coverage Path Planning for Multi-Agent Systems with ACO and GA. In *2024 32nd International Conference on Electrical Engineering (ICEE)*, (pp. 1–6). IEEE. https://doi.org/10.1109/ICEE63041.2024.10667967

Bazi, S., Benzid, R., Bazi, Y., & Rahhal, M. M. A. (2021). A Fast Firefly Algorithm for Function Optimization: Application to the Control of BLDC Motor. *Sensors*, 21(16), 5267. https://doi.org/10.3390/s21165267

Biswas, S., Anavatti, S. G., & Garratt, M. A. (2017). Particle swarm optimization based co-operative task assignment and path planning for multi-agent system. In 2017 IEEE Symposium Series on Computational Intelligence (SSCI), (pp. 1–6). IEEE. https://doi.org/10.1109/SSCI.2017.8280872

Dhulkefl, E. J., & Durdu, A. (2019). Path Planning Algorithms for Unmanned Aerial Vehicles. *International Journal of Trend in Scientific Research and Development*, 3(4), 359–362. https://doi.org/10.31142/ijtsrd23696

Dorigo, M. (1992). *Optimization, Learning and Natural Algorithms*. Ph.D. Thesis, Politecnico di Milano. https://cir.nii.ac.jp/crid/1573950400977139328

Eberhart, R., & Kennedy, J. (1995). A new optimizer using particle swarm theory. In *Proceedings of the Sixth International Symposium on Micro Machine and Human Science* (pp. 39–43). IEEE. https://doi.org/10.1109/MHS.1995.494215

El-Shorbagy, M. A. (2026). A Review of Genetic Algorithms: Principles, Procedures, and Applications in Optimization. *Computer Modeling in Engineering & Sciences*, 147(1), 1–10. https://doi.org/10.32604/cmes.2026.079859

Evaznia, N., & Ebrahimi, R. (2023). Providing a Solution for Optimal Management of Resources using the Multi-objective Crow Search Algorithm in Cloud Data Centers. In *2023 9th International Conference on Web Research (ICWR),* (pp. 179–184). IEEE. https://doi.org/10.1109/ICWR57742.2023.10139192

Gharehchopogh, F. S., & Gholizadeh, H. (2019). A comprehensive survey : Whale Optimization Algorithm and its applications. *Swarm and Evolutionary Computation*, 48, 1–24. https://doi.org/10.1016/j.swevo.2019.03.004

Grenouilleau, F., Hoeve, W.-J. V., & Hooker, J. N. (2019). A Multi-Label A* Algorithm for Multi-Agent Pathfinding. In *Proceedings of the International Conference on Automated Planning and Scheduling*, (pp. 181–185). Association for the Advancement of Artificial Intelligence. https://doi.org/10.1609/icaps.v29i1.3474

Guntsch, M., & Middendorf, M. (2002). Applying Population Based ACO to Dynamic Optimization Problems. In M. Dorigo, G. Di Caro, & M. Sampels (Eds.), *Ant Algorithms* (pp. 111–122). Springer. https://doi.org/10.1007/3-540-45724-0_10

Gupta, A., & Srivastava, S. (2020). Comparative Analysis of Ant Colony and Particle Swarm Optimization Algorithms for Distance Optimization. *Procedia Computer Science*, 173, 245–253. https://doi.org/10.1016/j.procs.2020.06.029

Houssein, E. H., Saeed, M. K., Hu, G., & Al-Sayed, M. M. (2024). Metaheuristics for Solving Global and Engineering Optimization Problems: Review, Applications, Open Issues and Challenges. *Archives of Computational Methods in Engineering*, 31, 4485–4519. https://doi.org/10.1007/s11831-024-10168-6

Huang, S., Yang, D., Zhong, C., Yan, S., & Zhang, L. (2021). An Improved Ant Colony Optimization Algorithm for Multi-Agent Path Planning. In *2021 International Conference on Networking Systems of AI (INSAI),* (pp. 95–100). IEEE. https://doi.org/10.1109/INSAI54028.2021.00028

Hussien, A. G., Amin, M., Wang, M., Liang, G., Alsanad, A., Gumaei, A., & Chen, H. (2020). Crow Search Algorithm: Theory, Recent Advances, and Applications. *IEEE Access*, 8, 173548–173565. https://doi.org/10.1109/ACCESS.2020.3024108

Islam, J., Vasant, P. M., Negash, B. M., & Watada, J. (2019). A modified crow search algorithm with niching technique for numerical optimization. In *2019 IEEE Student Conference on Research and Development (SCOReD),* (pp. 170–175). IEEE. https://doi.org/10.1109/scored.2019.8896291

Jia, Y.-H., Qiu, J., Ma, Z.-Z., & Li, F.-F. (2021). A Novel Crow Swarm Optimization Algorithm (CSO) Coupling Particle Swarm Optimization (PSO) and Crow Search Algorithm (CSA). *Computational Intelligence and Neuroscience*, 2021(1), 6686826. https://doi.org/10.1155/2021/6686826

Jin, S., Wang, Y., Duan, P., Zhang, H., Li, G., & Cai, Z. (2025). Multiagent Confrontation Method Based on Three-Party Dynamic Multistrategy Evolutionary Game. *IEEE Transactions on Computational Social Systems*, 12(5), 2604–2617. https://doi.org/10.1109/TCSS.2025.3538849

Joshi, A. S., Kulkarni, O., Kakandikar, G. M., & Nandedkar, V. M. (2017). Cuckoo Search Optimization- A Review. *Materials Today: Proceedings*, 4(8), 7262–7269. https://doi.org/10.1016/j.matpr.2017.07.055

Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization. Technical Report–TR06. https://lia.disi.unibo.it/Courses/SistInt/articoli/bee-colony1.pdf

Król, D., & Drożdżowski, M. (2010). Use of MaSE methodology for designing a swarm-based multi-agent system. *Journal of Intelligent & Fuzzy Systems*, 21(3), 221–231. https://doi.org/10.3233/IFS-2010-0453

Kumar, V., & Kumar, D. (2021). A Systematic Review on Firefly Algorithm: Past, Present, and Future. *Archives of Computational Methods in Engineering*, 28(4), 3269–3291. https://doi.org/10.1007/s11831-020-09498-y

Laassami, F., Ledmi, A., & Souidi, M. E. H. (2026). CSO: A Novel Crow Swarm Optimization Algorithm for Collaborative Path Planning with Collision Avoidance. *Journal of Intelligent & Fuzzy Systems*, 50(8), 2471–2496. https://doi.org/10.1177/18758967261422625

Laassami, F., Souidi, M. E., & Ledmi, A. (2025). Metaheuristics for Multi-Agent Control: An Overview. In *2025 International Conference on Recent Advances in Mathematics and Informatics (ICRAMI).* IEEE. https://doi.org/10.1109/ICRAMI64946.2025.11472735

Laassami, F., Souidi, M. E. H., & Ledmi, A. (2024). IB-PSO: An inversed Butterworth particle swarm optimisation algorithm for multi-agent path planning. *International Journal of Systems, Control and Communications*, 15(4), 387–410. https://doi.org/10.1504/IJSCC.2024.143853

Lamini, C., Benhlima, S., & Elbekri, A. (2018). Genetic Algorithm Based Approach for Autonomous Mobile Robot Path Planning. *Procedia Computer Science*, 127, 180–189. https://doi.org/10.1016/j.procs.2018.01.113

Lipowski, A., & Lipowska, D. (2012). Roulette-wheel selection via stochastic acceptance. *Physica A: Statistical Mechanics and Its Applications*, 391(6), 2193–2196. https://doi.org/10.1016/j.physa.2011.12.004

Loisel, F. (2023). *Decentralized Task Coordination in Heterogeneous IoT Clusters*. Diploma Thesis, Technische Universität Wien. https://doi.org/10.34726/HSS.2023.105524

Long, J., Wu, S., Han, X., Wang, Y., & Liu, L. (2023). Autonomous Task Planning Method for Multi-Satellite System Based on a Hybrid Genetic Algorithm. *Aerospace*, 10(1), 70. https://doi.org/10.3390/aerospace10010070

Ma, X., Zhang, C., Yao, F., & Li, Z. (2022). Multi-Agent Task Allocation Based on Discrete DEPSO in Epidemic Scenarios. *IEEE Access*, 10, 131181–131191. https://doi.org/10.1109/ACCESS.2022.3228918

Majumder, C. G., Kumar, L. R., & Philip, N. K. (2016). Bee foraging inspired multi-agent optimal motion planning analysis in a simulated-mobile environment. In *2016 Indian Control Conference (ICC),* (pp. 39–46). IEEE. https://doi.org/10.1109/INDIANCC.2016.7441103

Mandala, J., & Chandra Sekhara Rao, M. V. P. (2019). Privacy preservation of data using crow search with adaptive awareness probability. *Journal of Information Security and Applications*, 44, 157–169. https://doi.org/10.1016/j.jisa.2018.12.005

Mas, I., & Kitts, C. (2010). Centralized and decentralized multi-robot control methods using the cluster space control framework. In *2010 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, (pp. 115–122). IEEE. https://doi.org/10.1109/AIM.2010.5695768

McArthur, S. D. J., Davidson, E. M., Catterson, V. M., Dimeas, A. L., Hatziargyriou, N. D., Ponci, F., & Funabashi, T. (2007). Multi-Agent Systems for Power Engineering Applications—Part I: Concepts, Approaches, and Technical Challenges. *IEEE Transactions on Power Systems*, 22(4), 1743–1752. https://doi.org/10.1109/TPWRS.2007.908471

Meraihi, Y., Gabis, A. B., Ramdane-Cherif, A., & Acheli, D. (2021). A comprehensive survey of Crow Search Algorithm and its applications. *Artificial Intelligence Review*, 54(4), 2669–2716. https://doi.org/10.1007/s10462-020-09911-9

Milano, M., & Roli, A. (2004). MAGMA: A Multiagent Architecture for Metaheuristics. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, 34(2), 925–941. https://doi.org/10.1109/TSMCB.2003.818432

Mirjalili, S., & Lewis, A. (2016). The Whale Optimization Algorithm. *Advances in Engineering Software*, 95, 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008

Moradi, P., Imanian, N., Qader, N. N., & Jalili, M. (2018). Improving exploration property of velocity-based artificial bee colony algorithm using chaotic systems. *Information Sciences*, 465, 130–143. https://doi.org/10.1016/j.ins.2018.06.064

Najm, H. T., Ahmad, N. S., & Al-Araji, A. S. (2024). Enhanced path planning algorithm via hybrid WOA-PSO for differential wheeled mobile robots. *Systems Science & Control Engineering*, 12(1), 2334301. https://doi.org/10.1080/21642583.2024.2334301

Nayeem, G. M., Fan, M., & Akhter, Y. (2021). A Time-Varying Adaptive Inertia Weight based Modified PSO Algorithm for UAV Path Planning. In *2021 2nd International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST),* (pp. 573–576). IEEE. https://doi.org/10.1109/ICREST51555.2021.9331101

Necira, A., Naimi, D., Salhi, A., Salhi, S., & Menani, S. (2022). Dynamic crow search algorithm based on adaptive parameters for large-scale global optimization. *Evolutionary Intelligence*, 15(3), 2153–2169. https://doi.org/10.1007/s12065-021-00628-4

Priya, A., & Sahana, S. K. (2022). An Optimized Modelling and Simulation on Task Scheduling for Multi-Processor System using Hybridized ACO-CVOA. *Journal of Information Science and Engineering*, 38(5), 895–907. https://doi.org/10.6688/JISE.202209_38(5).0001

Rath, A. K., Parhi, D. R., Das, H. C., Kumar, P. B., & Mahto, M. K. (2021). Design of a hybrid controller using genetic algorithm and neural network for path planning of a humanoid robot. *International Journal of Intelligent Unmanned Systems*, 9(3), 169–177. https://doi.org/10.1108/IJIUS-10-2019-0059

Saeed, R. A., Omri, M., Abdel-Khalek, S., Ali, E. S., & Alotaibi, M. F. (2022). Optimal path planning for drones based on swarm intelligence algorithm. *Neural Computing and Applications*, 34(12), 10133–10155. https://doi.org/10.1007/s00521-022-06998-9

Sahu, D., Sinha, P., Prakash, S., Yang, T., Rathore, R. S., & Wang, L. (2025). A multi objective optimization framework for smart parking using digital twin pareto front MDP and PSO for smart cities. *Scientific Reports*, 15(1), 7783. https://doi.org/10.1038/s41598-025-91565-0

Sampson, J. R. (1976). Adaptation in Natural and Artificial Systems (John H. Holland). *SIAM Review*, 18(3), 529–530. https://doi.org/10.1137/1018105

Sariff, N., & Buniyamin, N. (2006). An Overview of Autonomous Mobile Robot Path Planning Algorithms. In *2006 4th Student Conference on Research and Development,* (pp. 183–188). IEEE. https://doi.org/10.1109/SCORED.2006.4339335

Sharma, A., Srinivasan, D., & Kumar, D. S. (2016). A comparative analysis of centralized and decentralized multi-agent architecture for service restoration. In *2016 IEEE Congress on Evolutionary Computation (CEC),* (pp. 311–318). IEEE. https://doi.org/10.1109/CEC.2016.7743810

Sood, M., & Panchal, V. K. (2024). Optimal path planning with hybrid firefly algorithm and cuckoo search optimisation. *International Journal of Advanced Intelligence Paradigms*, 27(3/4), 223–248. https://doi.org/10.1504/IJAIP.2024.138565

Souidi, M. E. H., Haouassi, H., Ledmi, M., Maarouk, T. M., & Ledmi, A. (2023). A discrete particle swarm optimization coalition formation algorithm for multi-pursuer multi-evader game. *Journal of Intelligent & Fuzzy Systems*, 44(1), 757–773. https://doi.org/10.3233/JIFS-221767

Souidi, M. E. H., Ledmi, M., Maarouk, T. M., Ledmi, A., & Laassami, F. (2024). IMAP-QL: An improved multi-agent pursuit path-planning based on Q-learning. *International Journal of Systems, Control and Communications*, 15(2), 159–178. https://doi.org/10.1504/IJSCC.2024.138549

Souidi, M. E. H., Piao, S., & Li, G. (2017). Mobile agents path planning based on an extension of Bug-Algorithms and applied to the pursuit-evasion game. *Web Intelligence*, 15(4), 325–334. https://doi.org/10.3233/WEB-170369

Sun, B., & Niu, N. (2024). Multi-AUVs cooperative path planning in 3D underwater terrain and vortex environments based on improved multi-objective particle swarm optimization algorithm. *Ocean Engineering*, 311, 118944. https://doi.org/10.1016/j.oceaneng.2024.118944

Tisue, S., & Wilensky, U. (2004). NetLogo: A simple environment for modeling complexity. In *Proceedings of the International Conference on Complex Systems* (pp. 1–10).

Trujillo-Romero, F. (2022). Path planning using metaheuristics. *Visión Electrónica*, 16(1), 29–40.

Wang, S., Liu, Y., Qiu, Y., Zhang, Q., Huo, F., Huangfu, Y., Yang, C., & Zhou, J. (2022). Cooperative Task Allocation for Multi-Robot Systems Based on Multi-Objective Ant Colony System. *IEEE Access*, 10, 56375–56387. https://doi.org/10.1109/ACCESS.2022.3165198

Wang, Y., Yang, R. R., Xu, Y. X., Li, X., & Shi, J. L. (2021). Research on Multi-Agent Task Optimization and Scheduling Based on Improved Ant Colony Algorithm. *IOP Conference Series: Materials Science and Engineering*, 1043(3), 032007. https://doi.org/10.1088/1757-899X/1043/3/032007

Wei-feng Gao, San-yang Liu, & Ling-ling Huang. (2013). A Novel Artificial Bee Colony Algorithm Based on Modified Search Equation and Orthogonal Learning. *IEEE Transactions on Cybernetics*, 43(3), 1011–1024. https://doi.org/10.1109/TSMCB.2012.2222373

Wong, K. Y., & Komarudin. (2008). Parameter tuning for ant colony optimization: A review. In *2008 International Conference on Computer and Communication Engineering*, (pp. 542–545). IEEE. https://doi.org/10.1109/ICCCE.2008.4580662

Wooldridge, M. J. (2012). *An introduction to multiagent systems* (2. ed.). Wiley.

Xin, J., Chen, G., & Hai, Y. (2009). A Particle Swarm Optimizer with Multi-stage Linearly-Decreasing Inertia Weight. In *2009 International Joint Conference on Computational Sciences and Optimization*, (pp. 505–508). IEEE. https://doi.org/10.1109/CSO.2009.420

Yahia, H. S., & Mohammed, A. S. (2023). Path planning optimization in unmanned aerial vehicles using meta-heuristic algorithms: A systematic review. *Environmental Monitoring and Assessment*, 195(1), Article 30. https://doi.org/10.1007/s10661-022-10590-y

Yang, X.-S., & Deb, S. (2010). Cuckoo Search via Levy Flights. arXiv:1003.1594 [math.OC]. https://doi.org/10.48550/arXiv.1003.1594

Yazdanpanah, V., Dastani, M., Fatima, S., Jennings, N. R., Yazan, D. M., & Zijm, H. (2020). Task Coordination in Multiagent Systems. In *AAMAS '20: Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems,* (pp. 2056–2058). ACM.

Zheng, Y., Wang, L., & Xi, P. (2018). Improved Ant Colony Algorithm for Multi-Agent Path Planning in Dynamic Environment. In *2018 International Conference on Sensing,Diagnostics, Prognostics, and Control*, (pp. 732–737). IEEE. https://doi.org/10.1109/SDPC.2018.8664885

Zhu, X., & Wang, H. (2018). A new inertia weight control strategy for particle swarm optimization. *AIP Conference Proceedings*, 1955, 040095. https://doi.org/10.1063/1.5033759

Acta Informatica Pragensia is published by the Prague University of Economics and Business, Czech Republic | eISSN: 1805-4951