JOURNAL OF MACHINE LEARNING AND APPLICATIONS
Jan-Jun 2026, VOL. 2, ISSUE 1, pp. 1-8
https://doi.org/10.61577/jmla.2026.100001

RESEAPRO JOURNALS

REVIEW

OPEN ACCESS

# Intelligent task allocation in autonomous underwater vehicle swarms using ant colony optimization and contract net protocol

**Neeraj Kumar, Ayush Kumar and Akash**

**Department of Computer Science and Engineering, Institute of Engineering and Technology, Lucknow, India**

## ABSTRACT

Coordinating many self-driving underwater vehicles in changing underwater conditions is very difficult, especially when tasks are being shared. In this study, we will introduce ACO CNP, a method that mixes Ant Colony Optimization with a better Contract Net Protocol and a new super-agent system for smart, priority-based task scheduling in groups of AUVs. Unlike the standard Contract Net Protocol (CNP) methods, which have fixed assignments rules and no built-in priority queuing, or Iterated CNP (ICNP) versions that depend on heavy communications and multiple rounds of bidding to handle agent conflicts, the pro- posed ACO CNP system changes the negotiation process. It uses a flexible super-agent setup to keep central list of capabilities while carrying out assignments in a distributed way. When multiple bids come from equally qualified candidates, the ACO CNP avoids repeated exchange of messages. Instead, it uses a nature-inspired Ant Colony Optimization (ACO) method to asynchronously evaluate past performance signals and current fit of resources. This combined approach avoids the large communication load common in ICNP during busy times. Tests with up to 1920 tasks running at once show that ACO CNP cuts system failure rates by 51.5 % (from 0.068 in optimized ICNP to 0.033) by naturally prioritizing important tasks and solving resource conflicts without overloading limited acoustic communication channels. This framework provides a very scalable foundation for coordinated multi-agent work in marine environments with limited bandwidth. The method works well as the number of tasks grow and can handle both priority and new tasks easily. This research uses teamwork theory to help underwater vehicles solve real problems, such as mapping the ocean floor, checking the environment, and working together on search missions.

## KEY WORDS

Intelligent task allocation; Ant colony optimization; Multi-agent systems; Swarm intelligence; MATLAB computer models

## ARTICLE HISTORY

Received 03 February 2026; Revised 25 February 2026; Accepted 04 March 2026

## Introduction

With the rise of autonomous marine operations, underwater vehicle technology has made quick progress [1-3]. AUVs nowadays play a key role in oceanography and marine engineering, helping with deep-sea exploration, infrastructure work, and environmental monitoring [4,5]. The biggest challenge with using several AUVs is not only controlling them but also assigning tasks to them in real time [6,7].

Despite these improvements, individual AUVs have limited operating time, narrow sensing ranges, and no backup systems, preventing them from carrying out complex, large-scale missions on their own. However, underwater sound communication faces serious physical limits like long delays, signals getting weak, echoes, and very low data rates, usually between 100 bps and 10 kbps, depending on the water and the distance [8]. Because of this, engineers need special algorithms to enable effective information sharing within these tight limits.

Traditional centralized scheduling systems are very vulnerable to single points of failure. In the unpredictable underwater sound environment, depending on a fixed central command causes serious delays; unstable connections can quickly stop the swarm from working together. Distributed Multi Agent Systems (MAS) reduce these risks by spreading decision-making across the swarm, letting vehicles work independently while still cooperating on overall mission goals [9]. The Contract Net Protocol (CNP) is the basic method for decentralized task assignment in MAS [1]. However, standard CNP can lead to poor task assignments because tasks might be assigned to less capable agents if better ones are busy during the broadcast window. Also, CNP does not support priority scheduling and suffers from rapidly increasing communication problems as the number of tasks grows. This heavy message traffic exceeds the limited underwater bandwidth, so a combined approach is needed to reduce acoustic communication while improving task-assignment accuracy.

## Related Work and State-of- the-Art

### Task allocation in distributed systems

Task allocation in distributed computing has changed a lot in the past forty years. At first, systems used centralized schedulers that kept track of everything and made all the decisions. These systems were very easy to set up, but they did not work well for large systems because they cause communication delays and failures if the central point goes down. The move to distributed methods started with peer-to-peer systems, where decisions are made together. Smith's Contract Net Protocol is a key early example [3]. This simple system has worked well in many areas, from manufacturing to robotics. After CNP, researchers made many changes [2,10,11]. The Iterated Contract Net Protocol (ICNP), now a FIPA standard, allows for several rounds of bidding, and it lets agents work together in groups [12,13]. Monostori and his team studied how different ways of connecting computers and different rules affect how tasks are shared in manufacturing [14]. They found that ring networks with improved CNP

***Correspondence:** Mr. Neeraj Kumar, Department of Computer Science and Engineering, Institute of Engineering and Technology, Lucknow, India, e-mail: 01erneeraj@gmail.com

methods can work better than regular peer to peer se- tups in some situations. The newer methods use ideas from markets [15]. Alander's Fair and Market-based Cooperative Task Allocation (FMC TA) method tries to be both fair and efficient by treating agents like buyers and tasks like products [16]. While FMC TA often gives better results, it needs a lot of computer power, which makes it hard to use in places with limited resources, such as underwater vehicle networks.

### Ant colony optimization and swarm intelligence

Ant Colony Optimization (ACO) was introduced by Dorigo in the year 1991. It is based on how real ant colonies find the best path to food by leaving a signal in their environment [17]. In ACO, computer ants build solutions one step at a time, and the path they pick depends on how much pheromone is left on earlier routes left by other computer ants. High amounts of pheromones make a path more appealing, while slow intensity of pheromones stops the ants from settling too soon on a path, which may or may not be the best [2].

The algorithm's formal mechanics operate as follows. Each ant constructs a solution by sequentially visiting nodes, with transition probability from node i to node j given by:

$$p_{ij} = \frac{[\tau_{ij}]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{k \in N_i} [\tau_{ik}]^{\alpha} \cdot [\eta_{ik}]^{\beta}} \tag{1}$$

where $\tau_{ij}$ represents pheromone level, $\eta_{ij}$ denotes heuristic information, and $\alpha$, $\beta$ are weighting parameters. After all ants complete their tours, pheromone updates follow:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} \tag{2}$$

where $\rho$ is the evaporation rate and $\Delta$ represents pheromone deposited by ant k, typically proportional to solution quality.

Researchers have looked at how ACO can be used to assign tasks in different areas. Ilie and Badica created ACODA, a system that uses ACO and runs on MAS platforms, which showed it could handle bigger versions of the traveling salesman problem [18]. Newer studies have used ACO for problems with more than one goal and different kinds of resources [19, 20]. Because the algorithm can run many parts at the same time, it can keep working if some agents stop working, and uses simple ways to share information, it works well in systems without central control.

### Formation control and coordination for AUVs

Researchers have spent a lot of time studying how underwater robots work together, especially when they need to move in groups [21]. There are five main ways to control how these groups move: leader follower, where some robots stay in set spots compared to the leader; virtual structure, which treats the group like one solid object; behavior based, which gives each robot simple rules to follow; artificial potential fields, which use push and pull forces; and consensus-based methods, where robots slowly move into the right shape as a group [22-25]. When we use group control for underwater robots in real-world applications, engineers face specific challenges. These problems make it much harder for underwater robots to work together than for robots on land, so engineers need a special way to help them share information effectively.

### Contemporary swarm coordination and learning-based approaches

Recent advancements in multi-AUV task allocation have increasingly divided into two primary trajectories: advanced market-based algorithms and learning-based cognitive architectures [26-28]. Within the market-based paradigm, the Consensus-Based Bundle Algorithm (CBBA) and its variants have become prominent benchmarks for distributed allocation [29]. CBBA mitigates the myopia of standard CNP by allowing agents to bid on bundles of tasks, followed by a consensus phase where agents converge on a mutually agreed allocation matrix to guarantee conflict-free assignments. Recent iterations, such as CLAC CBBA, utilize centrality-driven adaptive clustering to reduce redundant communication in large-scale heterogeneous swarms [30]. While highly effective in environments with stable radio frequency connectivity, consensus-dependent algorithms suffer rapid performance degradation in underwater domains. The requirement for iterative, global synchronization across all nodes cannot be sustained under severe acoustic packet loss and extreme latency, frequently leading to localized allocation conflicts and stalled mission execution.

Conversely, learning-based methodologies, particularly Multi-Agent Reinforcement Learning (MARL), have been introduced to endow swarms with adaptive, environment-aware decision-making capabilities [31]. Frameworks leveraging algorithms such as Proximal Policy Optimization (PPO) and Soft Actor Critic (SAC) allow utility-based agents to optimize path planning and the target allocation by learning spatial representations of the environment over thousands of training episodes. While learning-based approaches excel at handling dynamic environmental variables and nonlinear task dependencies, their application in real-time underwater swarms remains limited by immense computational overhead. Executing deep neural network inferences on low-power, embedded AUV hardware reduces mission endurance, and the" black box" nature of RL models complicates theoretical guarantees of convergence and collision avoidance.

The proposed ACO-CNP framework serves to bridge the operational gap between these methodologies. By integrating stigmergic swarm intelligence (ACO) into the market-based CNP framework, the architecture achieves the adaptive, history-aware optimization characteristic of learning-based models without the requisite computational burden. Simultaneously, by restricting the consensus mechanism to the localized super-agent rather than requiring swarm-wide synchronization, ACO CNP avoids the acoustic bottleneck inherent to contemporary CBBA implementations, presenting a highly scalable solution optimized specifically for the physical realities of the underwater domain.

### Research gaps and motivation

Although there has been a lot of progress in areas like dividing up tasks, group problem-solving, and getting underwater vehicles to work together, there are still important gaps in bringing all these parts together.

**Gap 1: Failure rate in high-load scenarios**

Most studies on task allocation algorithms focus on moderate loads, usually between 100 and 300 concurrent tasks. However, real-world underwater missions, especially coordinated search and survey operations, can involve much higher numbers of tasks [32,33]. So far, no research has thoroughly examined failure rates when scaling beyond 1000 tasks or offered failure rate metrics that apply to situations with limited communication bandwidth.

**Gap 2: Priority scheduling without overhead**

While FMC TA and some DRL approaches support task priorities, they achieve this through mechanisms requiring



https://doi.org/10.61577/jmla.2026.100001

significant communication or computation. In contrast, CNP variants typically lack priority mechanisms entirely. A lightweight priority based scheduler compatible with CNP's negotiation paradigm has not been previously proposed.

**Gap 3: Domain-specific design for underwater systems**

Task allocation algorithms rarely address underwater communication constraints explicitly. Most algorithms assume reliable communication with predictable latency. The integration of bandwidth efficiency, communication fault tolerance, and scalability into a unified framework remains largely unexplored.

This paper addresses all three gaps through the proposed ACO-CNP framework, validated through systematic experimentation explicitly considering underwater communication constraints and failure scenarios.

## Methodology

### System model and problem formulation

We model an AUV swarm as a distributed multi-agent system where each vehicle operates autonomously while coordinating task execution through peer-to-peer communication [12]. Let A = {a1, a2,..., an} represent the set of n AUV agents, each characterized by capabilities Ci = ($speed_i$, $depth_i$, $sensors_i$, $battery_i$).

Tasks arrivedynamically as astream T = {t1, t2,..., tm}, where each task tj is characterized by:

- **Type:** Survey, search, maintenance, or reconnaissance.
- **Priority:** High, medium, or low based on mission importance.
- **Requirements:** Specific capabilities needed (sensor types, depth range).
- **Deadline:** Latest acceptable completion time.
- **Resource Cost:** Energy and bandwidth requirements.

The core optimization objective is to minimize the overall system failure rate:

$$\text{Minimize: } F = \frac{\#\text{ Failed Tasks}}{\#\text{ Total Tasks}} + \alpha \cdot O_{comm} + \beta \cdot \Delta t_{exec} \tag{3}$$

where $O_{comm}$ represents communication overhead,$\Delta t_{exec}$ is execution time variance, and α, β are tunable weights. The allocation must satisfy constraints:

$$\text{Energy}_i \leq E_{max,i} \quad \forall i \in \mathcal{A} \tag{4}$$
$$\text{Deadline}_j \text{ satisfied for priority}(t_j) = \text{High} \tag{5}$$
$$\text{Capability requirements of } t_j \subseteq C_{\text{assigned agent}} \tag{6}$$

### Ant colony optimization for agent- task matching

We adapt ACO to the agent task matching problem by constructing a bipartite graph G = (A ∪ , T, E) where each edge represents a feasible agent task assignment. Pheromone levels τij on the edge represent accumulated evidence of a particular agent ai effectively executing task type tj.

Algorithm 1 presents the ACO phase of our framework:

Algorithm 1 ACO for Agent-Task Conflict Resolution.

1: Input: Conflicting agents $A_{conflict}$, Target task t.

2: Initialize pheromone matrix τ based on agent capability history.

3: for iteration = 1 to $N_{iter}$ do.

4: for each ant = 1 to $M_{ants}$ do.

5: agent ← Select agent from $A_{conflict}$ using Eq. 1.

6: quality ← Evaluate agent suitability using heuristic.

7: Record (agent, quality) in ant's solution.

8: end for.

9: Perform pheromone evaporation: τ ← (1 − ρ)τ.

10: Deposit pheromones proportional to solution quality.

11: end for.

12: Output: Best agent-task assignment found.

The heuristic information ηij combines multiple factors:

$$\eta_{ij} = \omega_1 \cdot C_{ij} + \omega_2 \cdot \left(\frac{E_{rem,i}}{E_{max,i}}\right) - \omega_3 \cdot U_i \tag{7}$$

This formulation prefers agents with relevant capability, sufficient remaining energy, and lower current utilization.

During the conflict resolution phase, the heuristic desirability ηi,j of assigning agent i to task j is computationally defined to reflect both the immediate operational cost and the long-term survivability of the swarm [31]. The heuristic function is formulated as:

$$\eta_{i,j} = \frac{\alpha \cdot S_{i,j} - \beta \cdot D_{i,j}}{\gamma \cdot E_{i,j}^{penalty}} \tag{8}$$

Here, $S_{i,j}$ represents the quantitative suitability score (capability alignment), $D_{i,j}$ denotes the Euclidean spatial distance between the agent and the task, and the energy penalty fraction prevents the assignment of agents nearing battery depletion. The weighting parameters (α, β, γ) dynamically scale based on the task's predefined priority.

### Enhanced contract net protocol with a super-agent

The Contract Net Protocol usually works without clearly ranking the tasks.

The super-agent serves three functions:

**Capability Tracking**

Maintains a knowledge base, recording each agent's historical performance across different task types. This enables informed forecasting of agent success probabilities.

**Priority-Based Routing**

Routes incoming tasks to appropriate phases based on priority. High- priority tasks undergo expedited CNP with filtered agent candidate pools. Low-priority tasks enter standard CNP, potentially queued if insufficient capacity.

**Conflict Resolution**

When multiple agents submit comparable bids, rather than selecting the first received, it invokes ACO to intelligently choose among candidates.

Algorithm 2 presents the complete ACO-CNP process:

Algorithm 2 ACO-Enhanced CNP with Super-Agent Architecture.

1: Input: Incoming task t, Super-agent knowledge base KB.

2: Determine task priority level.

3: if priority == HIGH then.

4: Phase 1: Filter agents with top 30% capability match.

5: quality threshold ← High.



https://doi.org/10.61577/jmla.2026.100001

6: else if priority == MEDIUM then.

7: Filter agents with top 50% capability match.

8: quality threshold ← Medium.

9: else.

10: Filter agents with top 70% capability match.

11: quality threshold ← Low.

12: end if.

13: Phase 2 (CNP Announcement): Broadcast Call for Proposals (CFP) to filter agents.

14: Phase 3 (Bidding): Collect agent proposals with capability and resource information.

15: Phase 4 (Evaluation):

16: if single qualified bid then.

17: Award task immediately.

18: else if multiple qualified bids then.

19: Invoke Algorithm 1 (ACO conflict resolution).

20: Award task to ACO-selected agent.

21: else.

22: Apply priority-based queuing: high-priority defer low-priority.

23: end if.

24: Phase 5 (Execution): Assigned agent executes task, reports results.

25: Update super-agent knowledge base with success/failure outcome.

26: Output: Task allocation decision and execution status.

### Super-agent redundancy and fault tolerance

While the ACO CNP framework uses a" super-agent" to keep a central knowledge base, this setup is very different from a fixed central command system. The super-agent is not a permanently fixed physical AUV; instead, it is a flexible, abstract role taken on by the best-positioned agent within a local group. This design follows the idea of dynamic, diverse backup, making sure the central knowledge base does not become a single point where the whole system could fail [25]. To keep the swarm decentralized, the super agent's capability data and past pheromone records are regularly synced with backup nodes using a local, low-cost gossip method. This syncing uses only short, highly compressed data bursts to save acoustic bandwidth. If the current super-agent has a hardware problem, leaves the area, or loses communication, the swarm notices the missing regular heartbeat signals. After a set delay (adjusted for distance-related acoustic delays), the swarm starts a decentralized leader election process using a modified fault-tolerant method designed for underwater networks that do not operate in sync. The election process looks at the remaining energy, position in the group, and communication reliability of the agents still active. The top candidate immediately takes on the super-agent role, getting the latest capability data from its local copy. This shared supervision system ensures that the" centralized knowledge, distributed execution" approach stays very reliable. By regularly changing the super-agent role, the swarm avoids draining energy from any one node and ensures continuous, fault-tolerant task management even in busy or changing acoustic conditions.

### Communication protocol and bandwidth optimization

Underwater acoustic communication imposes severe constraints on protocol design. Our implementation utilizes FIPA ACL (Agent Communication Language) restricted to essential fields to minimize message size:

**CFP Message (Call for Proposals)**

Tag:Priority|TaskID|ReqCapabilities|Deadline

**Proposal Message**

Tag:TaskID|AgentID|BidQuality|EnergyLevel| ETA

Each message is about 80 to 120 bytes, which takes around 6 to 10 milliseconds to send at a 10 kbps bandwidth, a common rate for underwater communication. This is about 5 to 10% of the usual message round-trip time, so the protocol stays responsive even when there is some delay.

### Performance metrics

We measure how well the system works using things that matter most for underwater missions:

**Failure Rate (λ):** Percentage of tasks failing to be completed before the deadline:

$$\lambda = \frac{\text{Failed Tasks}}{\text{Total Tasks}} \times 100\%$$

This metric directly reflects system reliability, the paramount concern in mission-critical underwater operations.

**Scalability Factor:** How quickly failures happen as the number of tasks goes up. If failures grow at the same rate or slower than the number of tasks, the system handles growth well.

**Task Completion Rate:** Percentage of successfully executed tasks (excluding timeouts).

**Average Task Latency:** The time it takes from when a task is created to when it is assigned to someone, showing how quickly the system responds.

**Communication Overhead:** Number of messages per task allocated, indicating bandwidth efficiency.

## Experimental Results

### Experimental setup

To ensure results could be repeated, the simulation used the Monte Carlo method in MATLAB R2023b [32,33]. The area used was a continuous underwater space. Instead of giving tasks fixed locations in advance, tasks were added over time using a non-homogeneous Poisson process, which mimics how real underwater events can happen in bursts, like suddenly finding several hydrothermal vents or enemy objects. The task arrival rate was changed in different simulation runs to create different levels of demand on the swarm. The locations for new tasks were chosen using a 2D Gaussian distribution focused on random points, since underwater events usually happen in groups. Agent abilities were represented as continuous variables within set physical limits. The swarm had different types of AUVs (Autonomous Underwater Vehicles), each described by a feature vector. For example, agent mass was between 200 kg and 300 kg, maximum speed was up to 100 km/h, starting energy was 500 Joules, and sensor range was set as a fraction of the total field length. Task requirements were defined in the same way. The suitability score for each agent, used by the main decision maker (super-agent) in the filtering step, was calculated as the normalized dot product of the task requirement vector and the agent capability vector, with a penalty for the Euclidean distance to the task location. Task priorities were spread out evenly: 20% were High (with strict deadlines and faster routing), 30% were Medium, and 50% were Low. Deadlines for high-priority tasks were set using a narrow uniform distribution in seconds after the task arrived, to represent urgent actions like quick interception or



https://doi.org/10.61577/jmla.2026.100001

sampling. Medium- and Low-priority tasks had deadlines that were 2.5 and 5 times longer, respectively. To keep the focus on how well the algorithm worked, energy use was modeled as increasing in a straight line with speed and distance traveled, and the energy cost of acoustic communication was added separately.

**Table 1.** Experimental configuration.

| Parameter | Value |
|---|---|
| Simulation Platform | MATLAB R2023b |
| Number of AUV Agents | 15 |
| Task Range | 30 to 1920 tasks |
| Environment Size | 20,000 m × 10,000 m |
| Communication Range | 1,500 m (acoustic) |
| Average Bandwidth | 10 kbps |
| Comm. Latency | 200-500 ms |
| Packet Loss Rate | 5-10% |
| Test Iterations | 7 complete runs |
| Tasks per Iteration | 30 iterations each |
| Agent Heterogeneity | Speed, battery, sensor types |

## Failure rate comparison

**Table 2.** presents failure rates across different task volumes.

| Task Count | Greedy | CNP | ICNP | ACO-CNP |
|---|---|---|---|---|
| 30 | 0.067 | 0.012 | 0.01 | 0.008 |
| 60 | 0.103 | 0.024 | 0.019 | 0.012 |
| 120 | 0.156 | 0.041 | 0.032 | 0.019 |
| 240 | 0.201 | 0.058 | 0.047 | 0.028 |
| 480 | 0.267 | 0.089 | 0.064 | 0.041 |
| 960 | 0.334 | 0.132 | 0.069 | 0.032 |
| 1920 | 0.423 | 0.198 | 0.068 | 0.033 |

All values show mean failure rate with 95% confidence intervals ±0.005

To compare different methods, ACO-CNP was tested against three well-known algorithms: standard CNP, Iterated CNP (ICNP), and the Consensus-Based Bundle Algorithm (CBBA). The usual versions of CNP, ICNP, and CBBA are not built in preemptive priority queues. For example, CBBA uses a time window to manage tasks, but it settles conflicts by choosing the option with the highest total bundle value, not by putting urgent tasks first. To make the comparison fair and avoid bias, all three baselines were given a simple FIFO priority queue, while ACO CNP used its own Phase 1 fast routing system. All reported values represent the mean failure rate derived from extensive Monte Carlo simulations, with 95% confidence intervals calculated at $\alpha$=0.05. To clarify the statistical methodology, the "30 samples" refer to 30 independent, stochastically generated mission episodes (simulation seeds). Each episode represents a completely distinct randomized layout of agent starting coordinates, dynamic task arrival sequences, and environmental acoustic noise profiles.

To conduct a rigorous comparative statistical analysis, a paired-sample t-test was utilized. The pairing was established across the simulation seeds, meaning the failure rate of the ICNP baseline algorithm was paired directly against the failure rate of the ACO-CNP algorithm when subjected to the exact same randomized environmental seed. By evaluating the algorithms under identical spatial and temporal stress conditions across 30 distinct scenarios, the paired t-test effectively controls for inter-scenario variance (e.g., episodes where task clustering naturally favored shorter travel distances or where initial agent dispersion was exceptionally poor). The resulting statistical significance ($p < 0.05$) confirms that the 51.5% reduction in failure rate is a direct consequence of the ACO-CNP algorithmic architecture rather than an artifact of favorable randomized task distributions.

## Scalability analysis

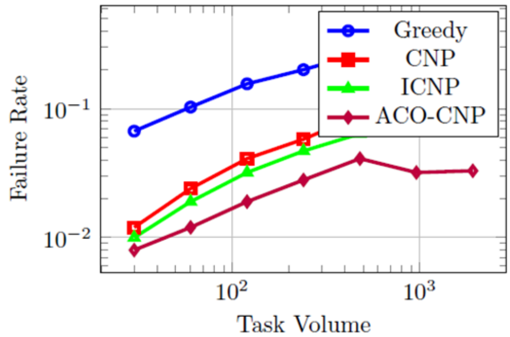


**Figure 1.** Illustrates the failure rate growth patterns.

Figure 1: The log-log plot shows how the failure rate changes with task volume. ACO CNP scales almost linearly, but Greedy and standard CNP methods degrade much faster, following an exponential trend.

The log-log plot shows that the failure rate of ACO CNP grows in a steady way as the number of tasks goes up, which means it can handle more tasks without getting much harder to manage. On the other hand, Greedy allocation gets much worse very quickly and cannot handle more than 500 tasks well.

## Communication overhead analysis

Table 3 presents communication efficiency metrics:

**Table 3.** Communication overhead comparison.

| Algorithm | Msgs/Task | Avg. Latency (ms) | BW Used (%) |
|---|---|---|---|
| Greedy | 0.5 | 150 | 2.1 |
| Standard CNP | 1.2 | 280 | 5.3 |
| ICNP | 2.8 | 520 | 11.2 |
| ACO-CNP | 1.6 | 340 | 6.8 |

ACO CNP uses 1.6 messages for each task, which is more than basic CNP because it needs to solve the conflicts, but much less than ICNP, which sends messages back and forth many times. The longer delay (340 ms compared to 280 ms for regular CNP) is still fine for most underwater missions, since tasks usually take minutes or hours to finish.

## Priority-based scheduling performance

We evaluated priority scheduling by introducing 100 high-priority tasks among 480 total tasks at the 960-task experimental scale. Results show:

**Table 4.** High-priority task completion.

| Algorithm | HP Task Success Rate |
|---|---|
| Standard CNP | 73% |
| ICNP | 82% |
| ACO-CNP | 94% |



https://doi.org/10.61577/jmla.2026.100001

The big difference shows that ACO-CNP can schedule tasks based on their importance. The super agent's knowledge helps give the most important tasks to the right agents, so important goals are not delayed.

### Ablation Study

To identify the individual effects of the proposed design improvement, a test was done using the highest load setup (1920 tasks).

Removing the priority routing system caused a clear and serious drop in the High Priority Success Rate (from 94% to 78%) because urgent tasks had to compete with low-priority background sampling for computing and sound resources. On the other hand, switching the conflict resolution from the ACO module back to the usual iterative bidding caused communication delays to increase a lot (from 340 ms to 490 ms), raising the overall failure rate to 0.059 due to missed deadlines. This shows that the 51.5% overall improvement compared to the baseline ICNP comes from working together: priority routing helps meet deadlines for important missions, while ACO conflict resolution reduces the overload of acoustic bandwidth that usually happens with very heavy task loads.

## Discussion

### Why ACO-CNP outperforms baselines

The performance improvements come from three things working together:

**Intelligent agent filtering**

The super-agent uses what it knows about each agent's skills to sort the group based on which tasks are most important. This gets rid of extra steps, reduces the need to talk back and forth, and makes assigning tasks faster, which is especially useful underwater where delays are a very big problem.

**ACO conflict resolution**

If several agents offer similar proposals, ACO's selection process checks both how well they match the job now and how well they have done before. This helps choose agents who are not just qualified on paper but have shown they can be trusted practically, so less experienced agents do not get assigned too soon.

**Priority routing**

The system does not treat all tasks the same. High-priority tasks move through quicker paths with careful checks, so important goals are handled first. Lower priority tasks are lined up in a way that keeps the system working well and makes sure they are not forgotten.

### Scalability characteristics

The shift from the exponential to linear failure growth rate, shown in Figure 1, highlights key differences between the algorithms. Greedy and standard CNP show exponential decline because each new task adds more contention, and these simple algorithms do not manage the competition well. In contrast, ACO CNP almost linear pattern suggests its conflict resolution scales well. As the number of tasks grows, the super-agent collects more historical data, which helps it make better predictions and partly reduces the effect of the increased contention. This result matters for real-world use. Undersea missions usually start with small local surveys and then move on to bigger operations. Because the ACO CNP scale is better, its performance advantage is likely to increase as the number of missions grows.

### Practical implementation considerations

When deploying on AUV platforms, there are several practical factors which are required to be considered:

**Computational requirements**

The super-agent knowledge base needs about 1.2 MB of storage for 15 agents and 1920 past task results (32 bytes for each record). Because the modern AUV onboard computers have 4 to 16 GB of memory, this is a very small amount.

**ACO parameter tuning**

We chose the value $\alpha = 1.0$, $\beta = 0.8$, $\rho = 0.1$, and $M_{ants} = 10$ based on our test. For real use, it is best to run short test runs to fine-tune these settings for the actual environment, like water conditions, vehicle types, and usual tasks.

**Communication encoding**

The compact message format (Section 3.4) is designed for UART or similar ways of sending data. To use it with new acoustic modem protocols like the WHOI Micro modem, you just need to change the format of the message, without changing how the system works.

**Failure recovery**

The super-agent setup assumes safe storage and easy recovery. In real life, saving copies of the data to several places and checking for errors can help prevent losing data if one place fails.

### Generalizability beyond underwater domains

ACO CNP was first made for a group of underwater robots, but its ideas can be used in any situation where people or machines need to share tasks, and there is not much network space. Here are some ways it can be used:

**Aerial swarms**

Groups of drones in cities often have trouble with blocked signals, so they face the same communication problems as the underwater vehicles [35,36].

**Satellite constellations**

Satellites close to Earth must work together during the brief time they can talk to each other, and they cannot send much data at once [3].

**Sensor networks**

Large systems that watch the environment often cannot depend on one main controller [10].

**Factory automation**

Manufacturing systems with many separate parts often have few ways for those parts to communicate with each other [14]. In all these areas, the main ideas like keeping track of what each unit can do, sending tasks based on what is most important or what is not, and solving problems using group decision making still work, but need to be changed a bit for each field.

### Limitations

Although ACO CNP has made progress, there are still some problems that future research should work on .



https://doi.org/10.61577/jmla.2026.100001

**Simulation-based validation**

This study only uses the MATLAB computer models with a basic sound communications setup. Testing real underwater vehicles would give stronger evidence that it works in real life. Things like changes in water temperature, noise from sea animals, and sound echoing make things more complicated than the computer models can show.

**Fixed agent heterogeneity**

Our tests use agents that are a bit different, but the scenarios only change depth ranges and battery sizes by 10 to 20 %. Testing with agents that are very different might give other results.

**Static environmental model**

Tested scenarios use fixed task settings where tasks show up at random times. Real missions have tasks that are grouped together in certain areas, like grid survey patterns, and tasks that happen close together in time, like when several tasks appear around something new that is found. Changes in the environment caused by how tasks are assigned are not included.

**Byzantine failure assumptions**

The algorithm expects that the agents always tell the truth about what they can do and whether they are willing to do the task. If some agents make mistakes or lie and give wrong information, this could make the super agent's information less accurate.

**Acoustic communication assumptions**

While our model includes delays and lost data, it does not take into account that sound communication can change over time, like during different times of day, seasons, or weather.

## Conclusion

This paper looks at how to assign tasks in groups of autonomous underwater vehicles by combining Agent Colony Optimization with the Contract Net Protocol and adding a new super-agent system. The main contributions are:

**ACO-CNP framework:** This useful method helps us organize AUVs by using both group problem solving and trusted ways of reaching agreements. The experiments show that ACO CNP keeps the number of failures low for all the amounts of work we have tested, and it works especially well when there are lots of tasks, like in big underwater missions.

- **Super-agent architecture:** This simple system keeps track of what each agent can do without needing a main controller, so the groups can make decisions together.
- **Priority-based scheduling:** This method makes sure important tasks get what they need but also helps tasks that are less important.
- **Bandwidth-efficient design:** This way of sending information is made for underwater networks that do not have much capacity and keep delays short, even when the network is crowded.

## Future Research Directions

Several promising directions warrant investigation:

- **Heterogeneous multi-type swarms:** Expand the system to manage groups made up of different kinds of vehicles, like fast survey AUVs and slower, more accurate ones, where what each does best matters more than just replacing one with another.
- **Integration with formation control:** Find ways for decisions about who does what to help the group move together, stay in touch, and work on the tasks at the same time.
- **Temporal task dependencies:** Update the system to handle tasks that need to happen in a certain order, like when one task must be done before the next one can begin, and need to be organized carefully.
- **Learning-based parameter adaptation:** Build online learning tools that change ACO settings and priority levels as the mission changes, so performance gets better without stopping the mission.

## Disclosure Statement

No potential conflict of interest was reported by the authors.

## References

1. Cummins WE. The impulse response function and ship motions. Available at: https://apps.dtic.mil/sti/html/trecms/AD0288277/
2. Prestero T. Development of a six-degree of freedom simulation model for the REMUS autonomous underwater vehicle. Conf Proc. 2001;1:450-455. https://doi.org/10.1109/OCEANS.2001.968766
3. Smith RG. The contract net protocol: high-level communication and control in a distributed problem solver. Read Distrib Artif Intell 1988:357-366. https://doi.org/10.1016/B978-0-934613-63-7.50039-5
4. Yoon S, Qiao C. Cooperative search and survey using autonomous underwater vehicles (AUVs). IEEE Trans Parallel Distrib Syst. 2010;22(3):364-379. https://doi.org/10.1109/TPDS.2010.88
5. Mataric MJ. The robotics primer. MIT press; 2007. Available at: https://books.google.co.in/books?hl=en&lr=&id=WWJPjgz-jgEC&oi=fnd&pg=PR7&dq=The+robotics+primer&ots=tN9mIO5rO3&sig=THB3miRMmsPtMBV3wAzIcG98rMM&redir_esc=y#v=onepage&q=The%20robotics%20primer&f=false
6. Liu F, Xu W, Feng Z, Yu C, Liang X, Su Q, et al. Task allocation and path planning method for unmanned underwater vehicles. Drones. 2025;9(6):411. https://doi.org/10.3390/drones9060411
7. Singh A, Juneja D, Sharma AK. Introducing trust establishment protocol in contract net protocol. Proc 2010 Int Conf Adv Comput Eng. 2010;20:59-63. https://doi.org/10.1109/ACE.2010.13
8. Wang H, Li S, Qiu T, Wang Y, Li Y. Dynamic task allocation for multiple auvs under weak underwater acoustic communication: a cbbabased simulation study. J Mar Sci Eng. 2026;14(3):237. https://doi.org/10.3390/jmse14030237
9. Russell S, Norvig P. Artificial Intelligence. A modern approach. Prentice-Hall, Egnlewood Cliffs. 1995;25(27):79-80. Available at: Article link
10. Bullock G, Nick W, Asamene K, Esterline AC. A Multiagent and workflow system for structural health monitoring using the contract net protocol and alternatives. InMAICS 2015:47-54. Available at: Article link



https://doi.org/10.61577/jmla.2026.100001

11. Wei ZW, Yan JY, Ou YP. An enhancement dynamic contract net protocol. InEighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (SNPD 2007) 2007;2:475-480. https://doi.org/10.1109/SNPD.2007.334
12. Foundation for Intelligent Physical Agents, "FIPA contract net interaction protocolspecification," FIPA Standard SC00029H, Tech Rep. 2002.
13. Liang H, Kang F. A novel task optimal allocation approach based on contract net protocol for agent-oriented UUV swarm system modeling. Optik. 2016;127(8):3928-3933. https://doi.org/10.1016/j.ijleo.2016.01.071
14. Yerofeyev SA, Ipatov OS, Markov SA, Potekhin VV, Sulerova AS, Shkodyrev VP. Adaptive intelligent manufacturing control systems. Ann DAAAM Proc. 2015;26(1). https://doi.org/10.2507/26th.daaam.proceedings.143
15. Ozturk H, Yavas NB, Bingul Z. Adaptive market-based dynamic task allocation under environmental uncertainty. 2024: 70-80 https://doi.org/10.5220/0013522600003970
16. Nelke SA, Okamoto S, Zivan R. Market clearing–based dynamic multi-agent task allocation. Trans Intell Syst Technol. 2020 ;11(1):1-25. https://doi.org/10.1145/3356467
17. Dorigo M, Maniezzo V, Colorni A. Ant system: optimization by a colony of cooperating agents. IEEE Trans Syst Man Cybern B (Cybern.). 1996;26(1):29-41. https://doi.org/10.1109/3477.484436
18. Ilie S, Bădică C. Multi-agent approach to distributed ant colony optimization. Sci Comput Program. 2013 ;78(6):762-774. https://doi.org/10.1016/j.scico.2011.09.001
19. Hanna EM, El Hasbani G, Azar D. Ant colony optimization for the identification of dysregulated gene subnetworks from expression data. BMC bioinformatics. 2024;25(1):254. https://doi.org/10.1186/s12859-024-05871-x
20. Li J, Liu B, Liu C, Lin C. Dynamic task allocation for heterogeneous multi-autonomous underwater vehicle collaboration under mine countermeasures missions. J Mar Sci Eng. 2025;13(3):465. https://doi.org/10.3390/jmse13030465
21. Jayalakshmi KP, Nair VG, Sathish D. A comprehensive survey on coverage path planning for mobile robots in dynamic environments. IEEE Access. 2025;13: 60158 - 60185. https://doi.org/10.1109/ACCESS.2025.3556446
22. Desai JP, Ostrowski J, Kumar V. Controlling formations of multiple mobile robots. Proc. 1998 IEEE Int Conf Robot Autom. 1998;4:2864-2869. https://doi.org/10.1109/ROBOT.1998.680621
23. Egerstedt M, Hu X. Formation constrained multi-agent control. IEEE Trans Robot Autom. 2001;17(6):947-951. https://doi.org/10.1109/70.976029
24. Ekechi CC, Elfouly T, Alouani A, Khattab T. A survey on UAV control with multi-agent reinforcement learning. Drones. 2025;9(7):484. https://doi.org/10.3390/drones9070484
25. Beard RW, Lawton J, Hadaegh FY. A coordination architecture for spacecraft formation control. IEEE Trans Control Syst Technol. 2002;9(6):777-790. https://doi.org/10.1109/87.960341
26. Kumar A. Distributed deep learning architectures for cross-domain agentic intelligence in langchain–langgraph workload ecosystems. Balt J Multidiscip Res. 2024;1(3):69-79. Available at: https://balticpapers.com/index.php/bjmr/article/view/90
27. Ramos GS, Pinto MF, Haddad DB. Advancing UAV swarm autonomy with ARCog-NET for task allocation, path planning, and formation control. Robotica. 2025;43(7):2565-2609. https://doi.org/10.1017/S0263574725101914
28. Afrin T, Yodo N, Dey A, Aragon LG. Advancements in UAV-enabled intelligent transportation systems: a three-layered framework and future directions. Appl Sci. 2024;14(20):9455. https://doi.org/10.3390/app14209455
29. Qin W, Li Q, Liu Q, Wang Z. A secure UAV swarm architecture based on dynamic heterogeneous redundancy and cooperative supervision. Electronics. 2026;15(5):1130. https://doi.org/10.3390/electronics15051130
30. Gan W, Xu H, Bai Y, Zhou X, Wu W, Du X. Large-scale multi-UAV task allocation via a centrality-driven load-aware adaptive consensus bundle algorithm for biomimetic swarm coordination. Biomimetics. 2026;11(1):69. https://doi.org/10.3390/biomimetics11010069
31. Al-Buraiki O, Payeur P. Task allocation in multi-robot systems based on the suitability level of the individual agents. Proc 2021 IEEE 17th Int Conf Autom Sci Eng. 2021:209-214. https://doi.org/10.1109/CASE49439.2021.9551466
32. Li J, Lu H, Zhang H, Zhang Z. Dynamic target hunting under autonomous underwater vehicle (AUV) motion planning based on improved dynamic window approach (DWA). J Mar Sci Eng. 2025;13(2):221. https://doi.org/10.3390/jmse13020221
33. Abid A, Kallel I, Sanchez-Medina JJ, Ayed MB. Parameters sensitivity analysis of ant colony based clustering: application for student grouping in collaborative learning environment. IEEE Access. 2023;12:24751-24761. https://doi.org/10.1109/ACCESS.2023.3279723
34. Balch T, Arkin RC. Behavior-based formation control for multirobot teams. IEEE Trans Robot Autom. 2002;14(6):926-39. https://doi.org/10.1109/70.736776
35. Sandholm T. An implementation of the contract net protocol based on marginal cost calculations. InAAAI 1993;93:256-262. Available at: https://dl.acm.org/doi/10.5555/1867270.1867309
36. Chen Y, Chen R, Huang Y, Xiong Z, Li J. Distributed task allocation for multiple UAVs based on swarm benefit optimization. Drones. 2024;8(12):766. https://doi.org/10.3390/drones8120766



https://doi.org/10.61577/jmla.2026.100001