UDC 004

I. Rudiakov, Ye. Batrak, N. Tsopa

COMPARATIVE EVALUATION OF TASK DISTRIBUTION ALGORITHMS IN MULTI-ROBOT SYSTEMS

Abstract: This article presents a comparative analysis of several task distribution algorithms applied in multi-robot systems. A set of classical, heuristic, and metaheuristic strategies was evaluated using simulation on randomly generated task sets with varying complexity and priority. Performance metrics such as ETA, robots queue size, and assignment time were analyzed to identify the most effective methods under different conditions.

Keywords: multi-robot systems, task distribution, algorithms, performance, optimization, robotics coordination, system, architecture, modeling, automatic control, robotic systems.

Introduction

Efficient task distribution is a core challenge in multi-robot systems, especially in environments with heterogeneous agents, dynamically changing workloads, and varying task complexities. Poor allocation can result in unbalanced robot loads, increased completion times, and inefficient use of resources.

While many systems rely on static or rule-based strategies, such methods often lack adaptability. To address these limitations, researchers proposed various intelligent algorithms that dynamically optimize task distribution.

This article presents a comparative evaluation of six task distribution algorithms: Least Connections, Load Balancing, Random, Ant Colony Optimization, Simulated Annealing, and Linear Optimization. These algorithms represent a diverse range of strategies – from simple heuristics to complex metaheuristics and mathematical optimization.

Each method was implemented in a custom task management platform and tested through simulations involving a fleet of 20 robots and 150-200 randomly generated tasks with different priorities, estimated durations, and constraints. Key evaluation metrics include average and maximum ETA (Estimated Time of Arrival), robot queue sizes, and average assignment time.

Related Work

Task allocation in multi-robot systems has been widely studied in recent decades, with approaches ranging from rule-based heuristics to complex optimization and learning-based models. Traditional algorithms such as Round Robin and Random assignment are simple and fast but often result in unbalanced workloads and inefficient system behavior.

More refined strategies like Least Connections and Load Balancing aim to distribute tasks based on current system load or robot availability. These are lightweight methods suitable for real-time systems but may struggle with task heterogeneity.

Metaheuristic approaches such as Ant Colony Optimization (ACO) and Simulated Annealing (SA) have shown promising results in solving NP-hard assignment problems by mimicking natural processes. ACO is inspired by the foraging behavior of ants and utilizes pheromone-based path reinforcement, while SA applies probabilistic search to escape local optima through gradual cooling of the solution space.

Mathematical models such as Linear Programming (LP) offer globally optimal solutions under certain constraints but often require a centralized controller and exact input parameters, which limits their real-time applicability.

Recent literature suggests that no single algorithm outperforms all others in every scenario. Instead, hybrid or adaptive approaches are increasingly proposed to exploit the strengths of different techniques depending on environmental conditions, task profiles, and robot capabilities.

Methodology

To evaluate and compare the performance of various task distribution algorithms in multi-robot systems, conducted a series of controlled simulations using a custom task assignment engine.

The testbed simulated a robot fleet of 20 agents and a randomly generated pool of 150-200 heterogeneous tasks, each characterized by a combination of:

1. Estimated Duration (ETA);
2. Priority level;
3. Task complexity;
4. Mandatory and optional capability requirements.

The following six algorithms were implemented and evaluated:

1. Least Connections – selects the robot with the fewest active tasks;
2. Load balancing – considers both the task queue size and cumulative task duration;
3. Random – assigns tasks to randomly selected capable robots;
4. Ant Colony Optimization (ACO) – simulates pheromone trails to identify optimal robots over multiple iterations;
5. Simulated Annealing (SA) – searches the solution space using a probabilistic acceptance of suboptimal solutions to avoid local minima;
6. Linear Optimization – selects the robot based on a weighted sum of its properties, optimized either for minimization of maximization depending on settings.

Each algorithm was tested under identical conditions using the same task and robot datasets. No algorithm had prior access to global task allocation outcomes, ensuring fair comparison based on real-time decision-making behavior.

Performance Metrics:

1. Average ETA per Robot – total expected time for each robot to complete its assigned queue;

 ISSN 1560-8956

2. Robot Pool Size – the number of active (uncompleted) tasks per robot;

3. Assignment Time – average computational time needed to assign a task;

4. Variance and Deviation – statistical dispersion of ETA and task count across all robots.

To analyze the performance of each task allocation strategy, a dedicated metrics collection and visualization subsystem was implemented as part of experimental infrastructure. All metrics were computed on the backend immediately after task distribution completion.

Metrics were retrieved through REST API endpoints exposed by the backend service. These endpoints returned detailed statistics for a given simulation run and selected algorithm. Each dataset included raw robot-level information, as well as aggregated statistical values, computed server-side.

The visualization of the collected data was handled entirely in the frontend. The dashboard was implemented in Angular and used the Apache ECharts library via “ngx-echarts” module to render dynamic charts. Upon receiving data from the server, the application parsed and transformed it into suitable chart series formats.

Three primary types of charts were generated:

1. ETA Distribution Chart – visualizing the total expected execution time per robot after full allocation;

2. Task Queue Size Chart – showing the number of assigned tasks per each robot (robot task pool size);

3. Average Distribution Time Chart – showing the average distribution time in seconds of each algorithm based on the allocation of all tasks.

This combination of backend-driven statistical aggregation and front-end-based chart rendering enabled fast iteration and reproducible testing across all task distribution strategies.

The visual component also played an essential role in communicating the results in a compact and accessible form, especially in comparing behavior between heuristic and deterministic algorithms.

This approach allowed for immediate and reproducible comparison of different algorithms across multiple simulation batches. It also ensured separation of concerns: the backend task evaluation and statistical aggregation, while the frontend focused on user interaction and visualization.

This modularity facilitated rapid development and adjustment of experiments with minimal effort.

The overview of algorithms

Least Connections

The Least Connections algorithm selects the robot with the smallest current task queue. It aims to balance task distribution evenly across all available agents, minimizing overload on individual robots.

Fig. 1 shows the distribution of Estimated Time of Arrival (ETA) per robot after full task allocation. The values exhibit relatively uniform distribution, with some spread due to varying tasks durations.

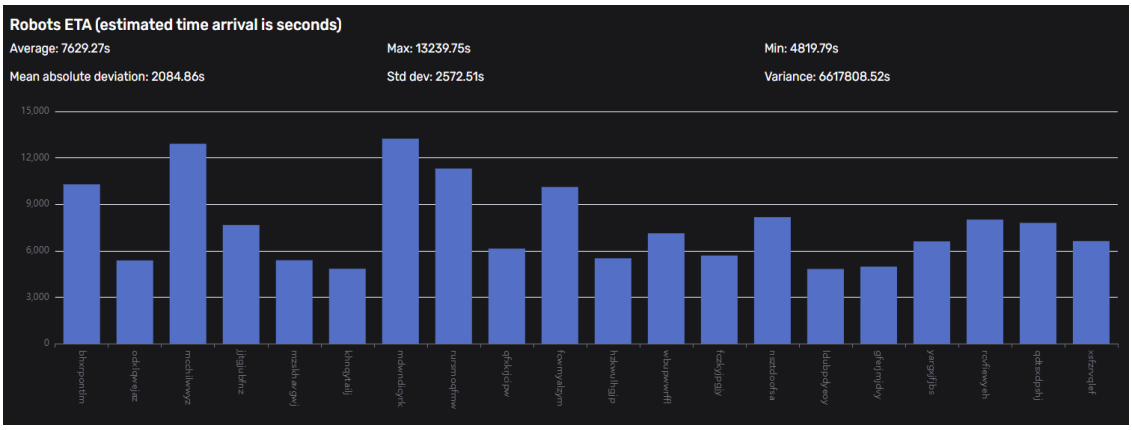

Figure 1. Robot ETA distribution for Least Connections

Fig. 2 illustrates the number of queued tasks for each robot. As expected, the algorithm achieves a near-uniform queue length across agents, ranging between 4 and 6 tasks.

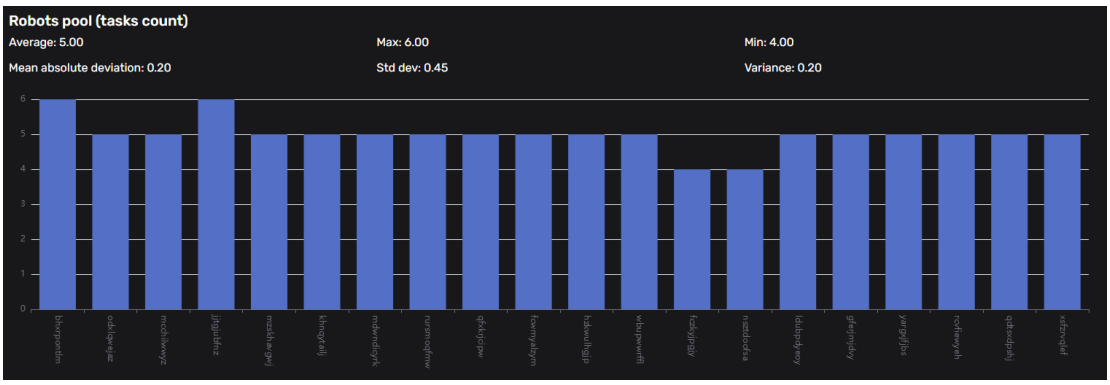

Figure 2. Robot task queue sizes for Least Connections

Tab. 1 summarizes the statistical performance metrics obtained during the experiment.

Table 1.

Performance metrics for Least Connections algorithm

Metric	ETA (s)	Tasks Queue (tasks)
Average	7269.60	5.00
Maximum	13239.75	6.00
Minimum	4819.79	4.00
Mean Absolute Deviation	2084.86	0.20
Standard Deviation	2572.51	0.45
Variance	6617808.00	0.20

The average assignment time for this algorithm was 430 ms, indicating fast execution suitable for real-time environments.

 ISSN 1560-8956

Summary: The Least Connections algorithm demonstrated strong load-balancing capabilities, maintaining narrow deviations in task queue sizes and ensuring consistent ETA distribution. However, its decision-making does not account for individual task complexities or robot capacities, which may affect performance in highly heterogeneous environments.

Load Balancing

The Load Balancing algorithm assigns tasks to the robot with the lowest cumulative workload, considering both the estimated duration and complexity of its currently queued tasks. This approach aims to more accurately reflect the true load on each robot, beyond just tasks count.

Fig. 3 displays the distribution of Estimated Time of Arrival (ETA) across all robots after task assignment. The results show a tightly grouped ETA range, indicating balanced scheduling across the fleet.

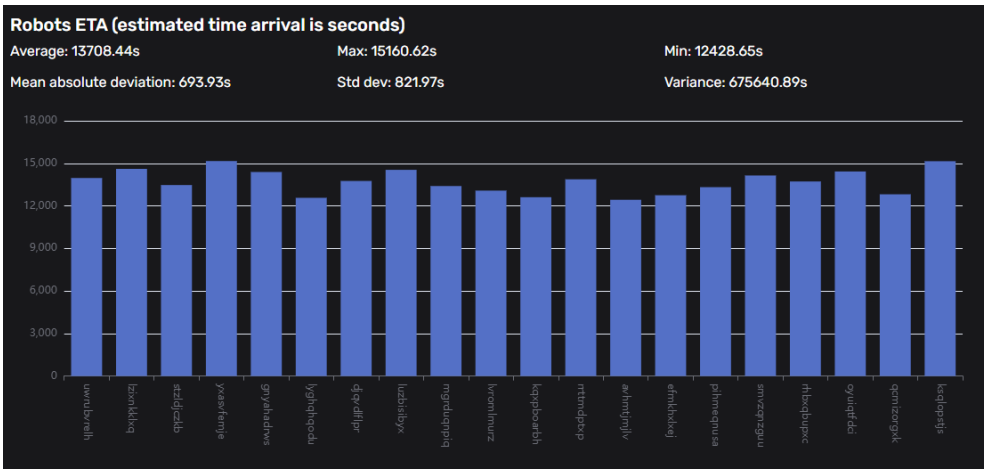

Figure 3. Robot ETA distribution for Load Balancing

Fig. 4 illustrates the number of queued tasks per robot. While the count ranges between 3 and 6, this is consistent with a strategy that prioritizes cumulative load rather than strict task counts.

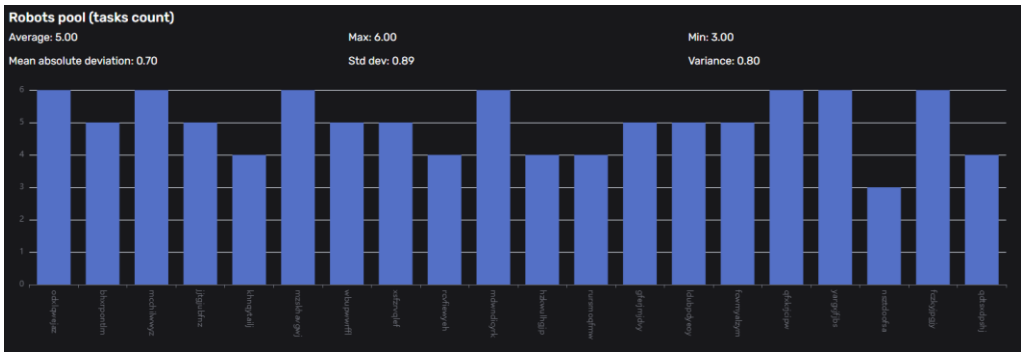

Figure 4. Robot task queue sizes for Load Balancing

Tab. 2 summarizes the statistical performance metrics obtained during the experiment.

Table 2.

Performance metrics for Load Balancing algorithm

Metric	ETA (s)	Tasks Queue (tasks)
Average	13708.44	5.00
Maximum	15160.62	6.00
Minimum	12428.65	3.00
Mean Absolute Deviation	693.93	0.70
Standard Deviation	821.97	0.89
Variance	675640	0.80

The average task assignment time was 420m ms, reflecting a high-performance implementation that remains efficient even with added complexity in decision-making.

Summary: Load Balancing offers an effective method of distributing workload evenly by accounting for task complexity and duration. It achieves good overall system balance and prevents overload on individual robots. The algorithm is particularly well-suited for environments with heterogeneous task profiles where fairness in execution time is critical.

Random

The Random algorithm assigns tasks to robots by randomly selecting an available agent from the filtered pool. It does not account for robot load, capabilities, or task complexity, making it the naivest of the considered approaches.

Fig. 5 illustrates Estimated Time of Arrival (ETA) distribution per robot after task allocation. The variance is much higher compared to more structured algorithms, with some robots having no load and others receiving large amounts of work.

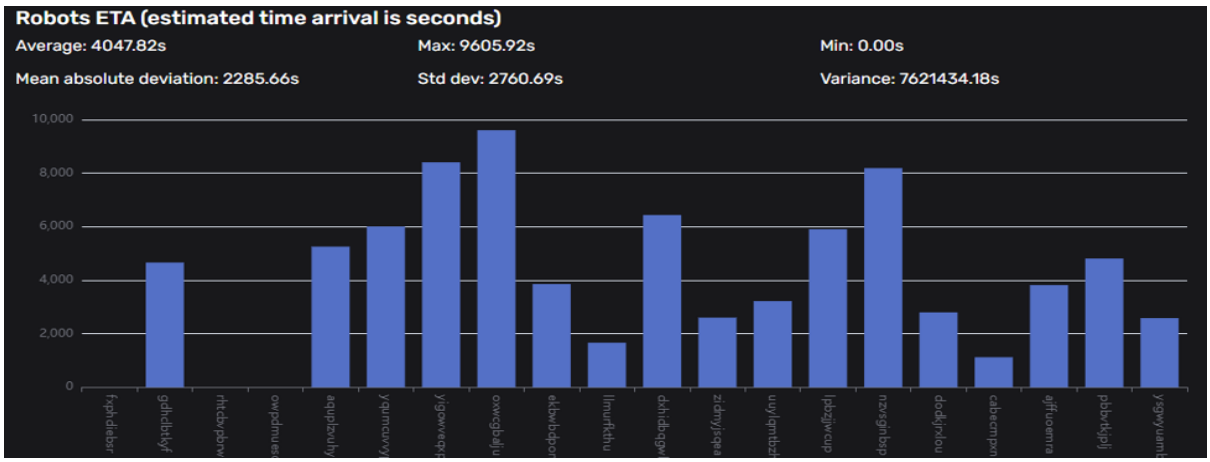

Figure 5. Robot ETA distribution for Random

Fig. 6 shows the number of tasks assigned to each robot. The range from 0 to 7 tasks indicates a highly uneven distribution, confirming the unregulated nature of this approach.

ISSN 1560-8956

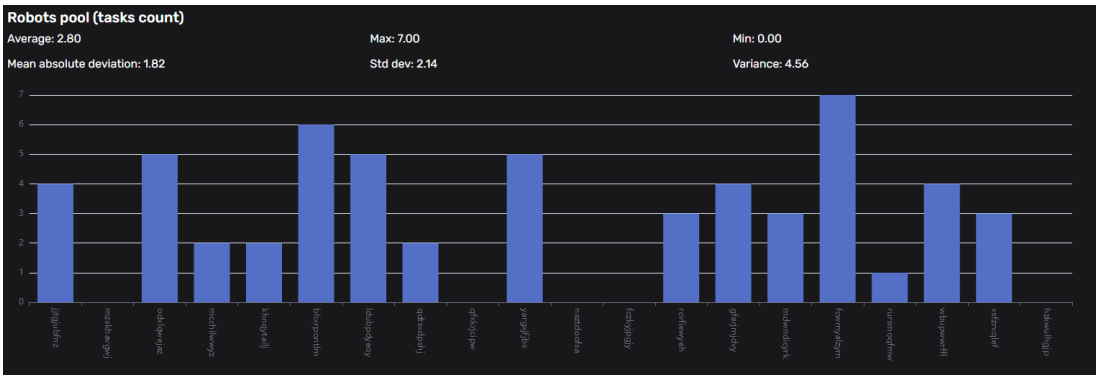

Figure 6. Robot task queue sizes for Random

Tab. 3 summarizes the statistical performance metrics obtained during the experiment.

Table 3.

Performance metrics for Random

Metric	ETA (s)	Tasks Queue (tasks)
Average	4047.82	2.80
Maximum	9605.92	7.00
Minimum	0.00	0.00
Mean Absolute Deviation	2285.53	1.82
Standard Deviation	2760.69	2.14
Variance	7621434	4.56

The average assignment time was 500 ms, reflecting the simplicity of the algorithm without computational overhead.

Summary: Random assignment results in highly unbalanced workload distribution, with wide ETA and task queue disparities across robots. Although computationally cheap and easy to implement, its unpredictability and inefficiency make it unsuitable for production environments requiring fairness or responsiveness.

Ant Colony

The Ant Colony Optimization (ACO) algorithm is a metaheuristic inspired by the foraging behavior of ants, where artificial agents (ants) explore assignment paths and reinforce successful ones through virtual pheromones. This approach aims to find a robot that best matches the task requirements through probabilistic selection influenced by both heuristic and accumulated pheromone strength.

Fig. 7 presents the ETA distribution across robots. The graph reveals significant variation, with a maximum value exceeding 25,000 seconds and several idle agents. Such variance indicates that while the algorithm can find optimal pairing, it may sometimes lead to unbalanced distributions when convergence is slow.

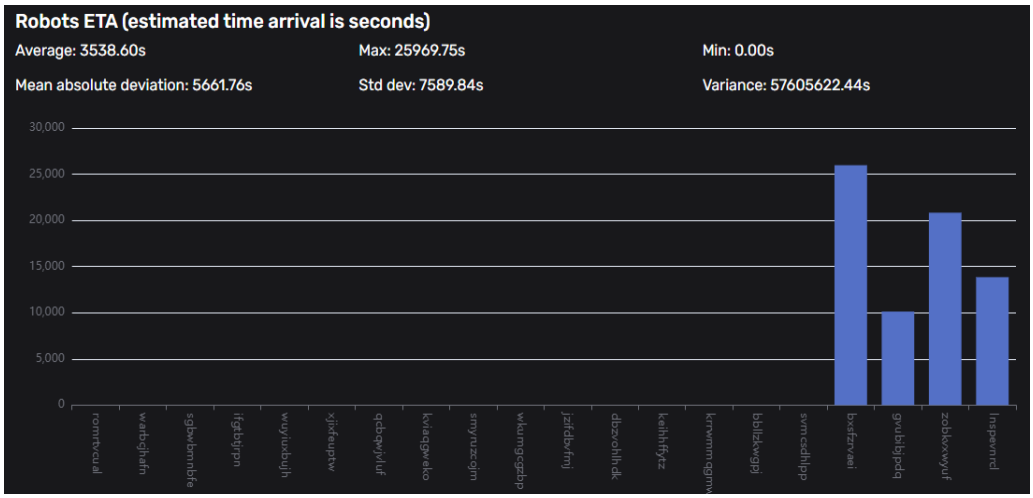

Figure 7. Robot ETA distribution for Ant Colony

Fig. 8 shows the number of tasks assigned per robot. The range spans from 0 to 15, confirming the uneven allocation pattern, which is a common trait of stochastic algorithms in non-homogeneous task environments.

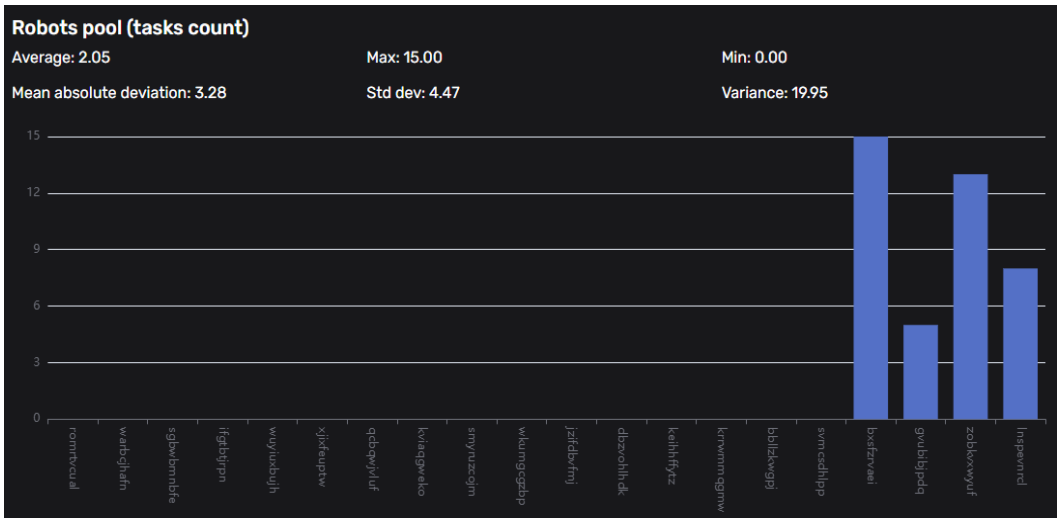

Figure 8. Robot task queue sizes for Ant Colony

Tab. 4 summarizes the statistical performance metrics obtained during the experiment.

The average task assignment time was 600 ms, due to iterate exploration and probability calculations.

Table 4.

Performance metrics for Ant Colony

Metric	ETA (s)	Tasks Queue (tasks)
Average	3538.60	2.05
Maximum	25969.75	15.00
Minimum	0.00	0.00
Mean Absolute Deviation	5661.76	3.28
Standard Deviation	7589.85	4.47
Variance	57605622	19.95

 ISSN 1560-8956

Summary: Ant Colony Optimization demonstrates potential for finding highly relevant robot-task matches, especially when tasks have diverse complexity or resource needs. However, the high variance in load and ETA suggests sensitivity to convergence parameters. This approach suits complex or dynamic system where adaptability is more valuable than strict load balancing.

Simulated Annealing

The Simulated Annealing (SA) algorithm is a probabilistic optimization method inspired by the annealing process in metallurgy. It explores the solution space by occasionally accepting suboptimal solutions to escape local minima. In context of task assignment, Simulated Annealing provides the flexibility to consider not only the most immediately optimal robots for a task, but also suboptimal candidates that might lead to better overall system performance in the long run. This can be especially valuable in dynamic or heterogeneous environments where robot states, capabilities, and workloads change frequently, and local optima may not reflect the best global decisions.

Fig. 9 illustrates the ETA distribution for all robots. Similar to other metaheuristic methods, the results exhibit a high degree of dispersion, with significant outliers. In this experiment, the maximum ETA exceeded 30,000 seconds, indicating that certain robots accumulated substantial task queues. The highlight the exploratory behavior of the algorithm, which can occasionally lead to imbalance in pursuit of broader optimization.

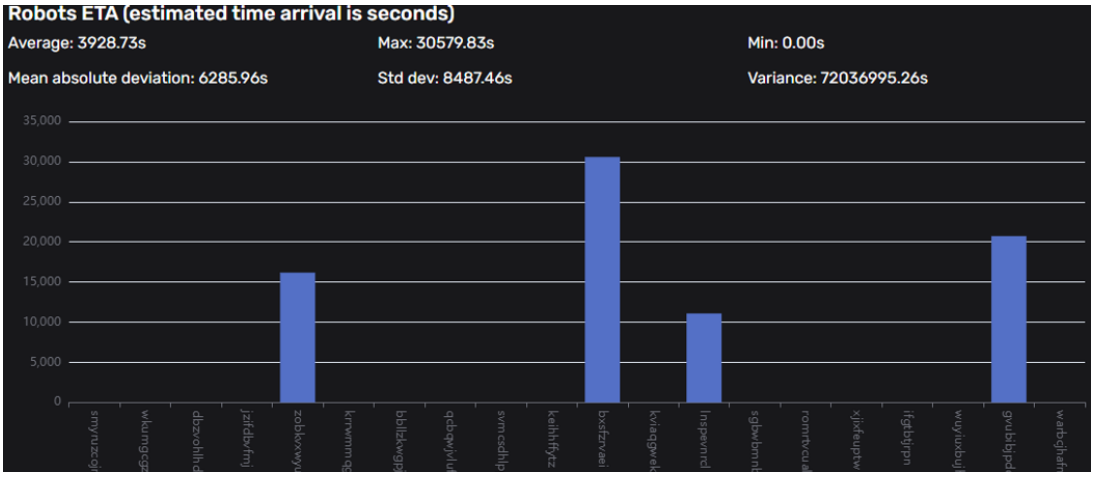

Figure 9. Robot ETA distribution for Simulated Annealing

Fig. 10 shows the number of tasks assigned to each robot. The queue sizes vary from 0 to 15 tasks, indicating variability in load distribution due to the acceptance of less optimal solutions during the search process.

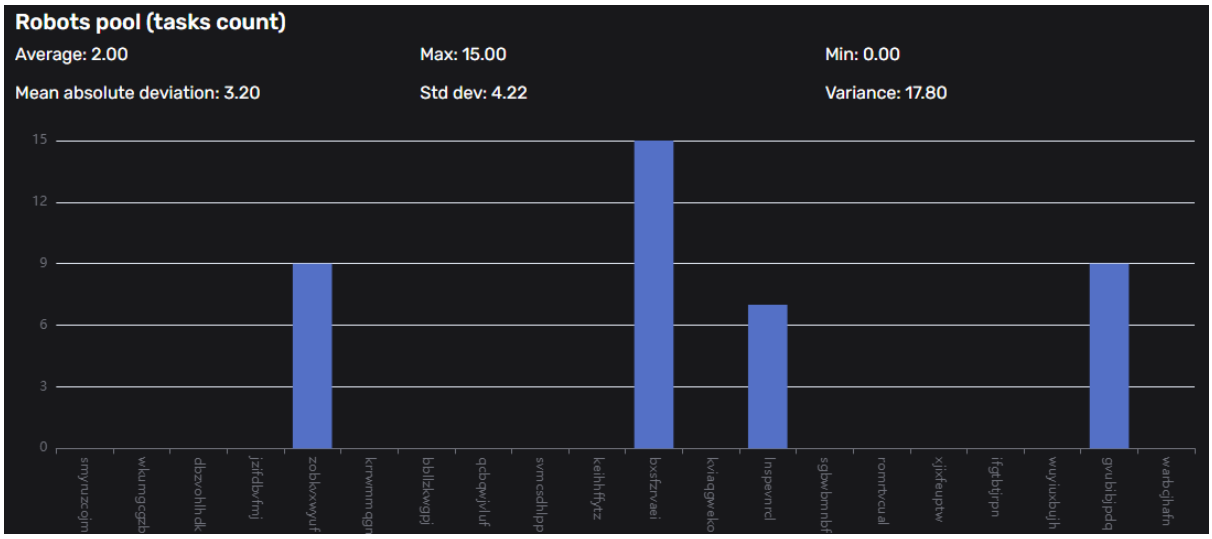

Figure 10. Robot task queue sizes for Simulated Annealing

Tab. 5 summarizes the statistical performance metrics obtained during the experiment.

Table 5.

Performance metrics for Simulated Annealing

Metric	ETA (s)	Tasks Queue (tasks)
Average	3928.73	2.00
Maximum	30579.83	15.00
Minimum	0.00	0.00
Mean Absolute Deviation	6285.96	3.20
Standard Deviation	8487.46	4.22
Variance	72036995	17.80

The average assignment time was 600 ms, mostly due to multiple iterations per temperature level and probability checks.

Summary: Simulated Annealing offers robust search capabilities, which can be useful in environments with unpredictable or complex task structures. Despite its flexibility, the algorithm tends to produce uneven results when convergence is not tightly controlled, making it better suited for experimental or adaptive task allocation scenarios rather than systems requiring strict real-time balancing.

Linear Optimization

The Linear Optimization algorithm is a mathematical approach that seeks to optimize a linear objective function under a set of linear constraints. In the context of task allocation for multi-robot systems, this method uses weighted robot properties and a defined goal (e.g., maximization or minimization) to select the most suitable executor. The optimization target is configured per robot category, allowing flexible behavior tailored to different robot groups or use cases.

 ISSN 1560-8956

Fig. 11 shows the Estimated Time of Arrival (ETA) distribution per robots. The results reveal a high variance across agents, with some robots heavily overloaded (maximum ETA exceeding 32,000 seconds), while others received no tasks at all. This suggests that while the algorithm can target optimal matches based on criteria, it may not prioritize balanced distribution.

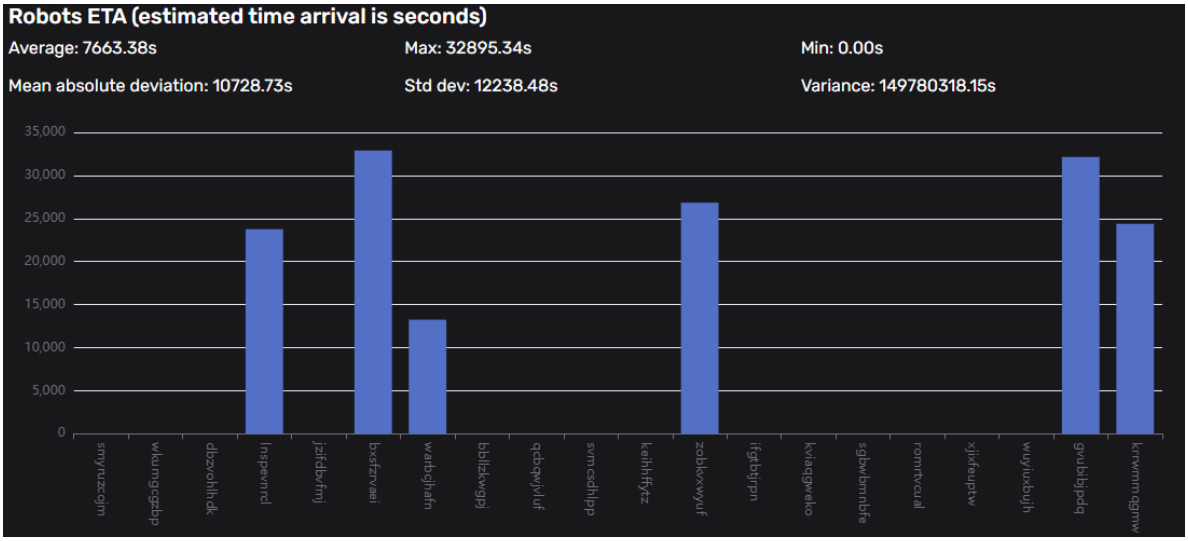

Figure 11. Robot ETA distribution for Linear Optimization

Fig. 12 presents the task queue sizes per robot after the allocation process. The spread is significant, ranging from 0 to 15 tasks, indicating that some robots were selected repeatedly while others remained idle. This unevenness is a byproduct of strict optimization rules focusing on maximizing score rather than distributing load fairly.

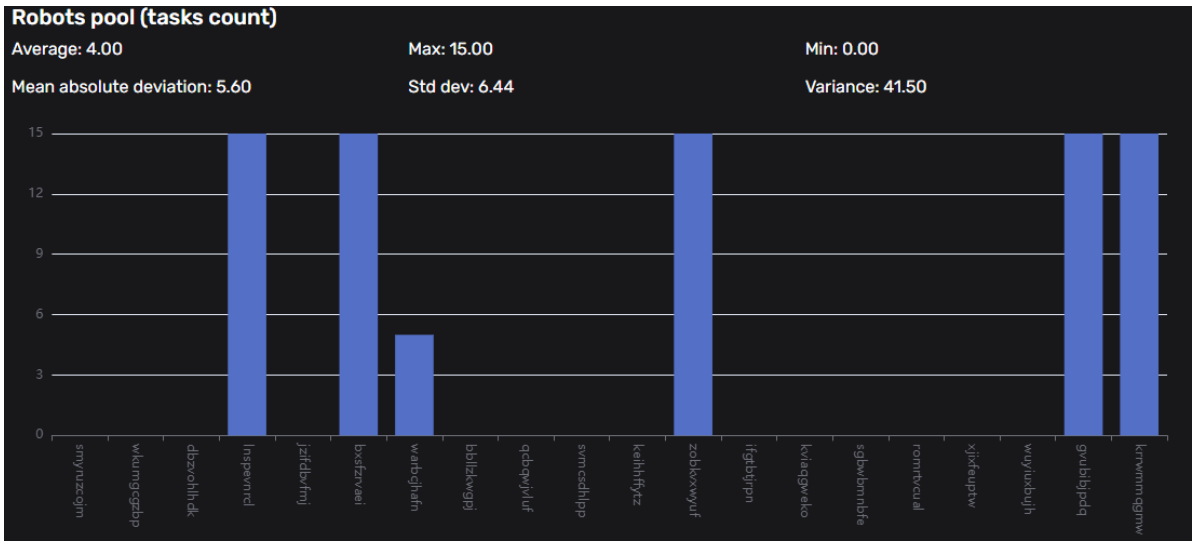

Figure 12. Robot task queue sizes for Linear Optimization

Tab. 6 summarizes the statistical performance metrics obtained during the experiment.

Table 6.

Performance metrics for Linear Optimization

Metric	ETA (s)	Tasks Queue (tasks)
Average	7663.38	4.00
Maximum	32895.34	15.00
Minimum	0.00	0.00
Mean Absolute Deviation	10728.73	5.6
Standard Deviation	12238.48	6.44
Variance	149780318	41.50

The average assignment time was 550 ms, showing good responsiveness, especially given the computational nature of the algorithm.

Summary: The algorithm offers precise and tunable selection based on predefined scoring models, making it useful in environments where specific robot characteristics are critical for task success. However, it lacks intrinsic mechanisms for load balancing, often resulting in large discrepancies in task distribution. For systems requiring fairness alongside optimality, additional balancing strategies may be necessary.

Results and discussion

The evaluation revealed significant differences in how the algorithms handle task distribution under identical conditions. The Least Connections algorithm provided the most balanced queues and consistent ETA values, making it well-suited for systems requiring predictable behavior.

Load Balancing, while slightly slower in terms of task completion, considered task duration and complexity, making it more adaptive in heterogeneous environments. However, it still showed moderate variance in performance.

The Random algorithm resulted in high ETA variance and uneven task queues, demonstrating its inefficiency for production scenarios but usefulness as a control baseline.

Metaheuristic algorithms like Ant Colony and Simulated Annealing achieved low average ETA values, showing their strength in finding good assignments under uncertainty. However, their stochastic nature led to unstable task queues and occasional overloads on individual robots.

Linear Optimization prioritized ideal robot-task matching, which led to extreme imbalances: some robots were overloaded, others remained idle. While powerful for maximizing individual fit, it failed to maintain system-wide load distribution.

Conclusion

This study presented a comparative evaluation of six task distribution algorithms within a simulated multi-robot system. Through controlled experiments, accessed each

 ISSN 1560-8956

method's suitability across several performance indicators. The findings underline the importance of aligning algorithmic design with operational goals: while heuristic methods may offer adaptability and efficiency, classical strategies ensure predictability and system-wide balance.

Future work

Future work could focus on hybrid strategies that combine the responsiveness of metaheuristics with the stability of deterministic approaches. Additionally, extending to real-world environments would provide further insights into robustness and scalability under dynamic condition.

REFERENCES

1. *Fang Z., Ma T., Huang J., Niu Z., Yang F.* Efficient Task Allocation in Multi-Agent Systems Using Reinforcement Learning and Genetic Algorithm. MDPI. URL: https://www.mdpi.com/2076-3417/15/4/1905 (date of access: 25.05.2025).

2. *Liu H., Zhang Y., Tang Y., Wang X.* A Novel Multi-Robot Task Allocation Model in Marine Plastics Cleaning Based on Replicated Dynamics. MDPI. URL: https://www.mdpi.com/2071-1050/15/6/5074 (date of access: 25.05.2025).

3. *Bronsdon C.* Multi-Agent Coordination Strategies: From Theory to Practice. Galileo. URL: https://www.galileo.ai/blog/multi-agent-coordination-strategies (date of access: 25.05.2025).

4. *Braquet M., Bakolas E.* Greedy Decentralized Auction-based Task Allocation for Multi-Agent Systems. URL: https://martinbraquet.com/wp-content/uploads/Greedy-Decentralized-Auction-based-Task-Allocation-for-Multi-Agent-Systems.pdf (date of access: 25.05.2025).

5. *Purwar A.* Automated Planning Tool Makes Work Order Allocation More Efficient. Amazon Science. URL: https://www.amazon.science/blog/automated-planning-tool-makes-work-order-allocation-more-efficient (date of access: 25.05.2025).

6. *Petzoldt C., Niermann D., Maack E., Sontopski M., Vur B., Freitag M.* Digital Twin-Based Task Allocation for Multi-Agent Systems in Production Logistics. MDPI. URL: https://www.mdpi.com/2076-3417/12/24/12645 (date of access: 25.05.2025).

7. Amazon's Robot Army Is Watching: New Tools Boost Fulfillment Vigilance. Tomorrow Desk. URL: https://tomorrowdesk.com/vigilance/amazon-robot-arm (date of access: 25.05.2025).

8. *Elmogy A. M.* Multi-Robot Task Allocation: A Review of the State-of-the-Art. URL: https://www.researchgate.net/publication/277075091_Multirobot_Task_Allocation_A_Review_of_the_State-of-the-Art (date of access: 25.05.2025).

9. Multi-Agent Coordination across Diverse Applications: A Survey. arXiv. URL: https://arxiv.org/html/2502.14743v1 (date of access: 25.05.2025).

10. Consensus-Based Fast and Energy-Efficient Multi-Robot Task Allocation. ScienceDirect.URL:https://www.sciencedirect.com/science/article/abs/pii/S0921889022001592 (date of access: 25.05.2025).

11. Task Allocation in Multi-Robot System Using Resource Sharing. PLOS ONE. URL: https://journals.plos.org/plosone/article?id=10.1371%2Fjournal.pone.0267982 (date of access: 25.05.2025).

12. *Yuan Y., Yang P., Jiang H., Shi T.* A Multi-Robot Task Allocation Method Based on the Synergy of the K-Means++ Algorithm and the Particle Swarm Algorithm. MDPI. URL: https://www.mdpi.com/2313-7673/9/11/694 (date of access: 25.05.2025).

13. *Patel R., Rudnick-Cohen E., Azarm S., Otte M., Xu H., Herrmann J. W.* A Comparative Study of Task Allocation Algorithms for Multi-Robot Systems. URL: https://user.eng.umd.edu/~jwh2/papers/Patel-ICRA-2020.pdf (date of access: 25.05.2025).