# Application of Ant Colony Optimization (ACO) to the Problem of Bin Packing

720032056
University of Exeter
Exeter, UK

**ACM Reference Format:**
720032056. 2024. Application of Ant Colony Optimization (ACO) to the Problem of Bin Packing. In . ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 Literature Review

The **Bin Packing Problem (BPP)** is a combinatorial optimization problem that involves packing a set of items, each with a given weight, into a fixed number of bins with limited capacity. The objective can vary, but it typically includes minimizing the number of bins used or distributing items in a way that balances the bin loads. Due to the **NP-hard** nature of BPP, finding exact solutions becomes computationally prohibitive for large problem instances [8]. As a result, heuristic and metaheuristic approaches, particularly nature-inspired algorithms, have gained traction for efficiently solving this problem [16].

### 1.1 Nature-Inspired Algorithms Overview

Nature-inspired algorithms mimic biological, evolutionary, and physical processes to find approximate solutions for complex optimization problems. These algorithms often utilize mechanisms like natural selection and swarm intelligence to explore large search spaces efficiently and avoid getting trapped in local optima. Popular nature-inspired algorithms applied to BPP include **Genetic Algorithms (GA)**, **Ant Colony Optimization (ACO)**, and **Particle Swarm Optimization (PSO)**, among others [18]. These approaches are attractive because they are adaptable, can be hybridized, and generally perform well on a wide variety of NP-hard problems, including BPP.

While traditional deterministic algorithms (e.g., greedy algorithms) tend to offer quick but often suboptimal solutions, nature-inspired algorithms provide a probabilistic mechanism that allows for exploration and exploitation of the solution space. This capability makes them more suitable for problems like BPP, where the search space is exponential, and traditional methods may falter [1].


*Conference'17, July 2017, Washington, DC, USA*

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

### 1.2 Genetic Algorithms (GA)

**Genetic Algorithms (GA)**, introduced by John Holland in the 1970s, simulate the process of natural selection to evolve a population of candidate solutions over time [10]. In GA, potential solutions are represented as individuals within a population, with each solution encoding a specific arrangement of items in bins for the BPP. The fitness of an individual is evaluated based on how well it balances bin loads or minimizes the number of bins used [9].

GA operates by iterating through a cycle of **selection**, **crossover**, and **mutation**. Selection favors individuals with higher fitness, allowing them to reproduce. Crossover combines genetic information from two parent solutions to generate offspring with mixed traits. Mutation introduces random changes to offspring, ensuring genetic diversity.

For BPP, a well-known variation of GA is the **Grouping Genetic Algorithm (GGA)**, introduced by Falkenauer [7]. GGA's solution representation is tailored specifically for BPP, ensuring that genetic operations respect the constraints of bin capacities. This approach has been successful in finding high-quality solutions for large BPP instances, outperforming other heuristics like **First-Fit Decreasing (FFD)** in many cases.

The adaptability of GA allows it to be customized for different variants of BPP. However, GA can suffer from slow convergence, especially if the population diversity diminishes over time, which may lead to premature convergence on suboptimal solutions. Careful parameter tuning (e.g., mutation rate and crossover probability) is often required to maintain a balance between exploration and exploitation [6].

### 1.3 Ant Colony Optimization (ACO)

**Ant Colony Optimization (ACO)** is a nature-inspired metaheuristic first introduced by Dorigo in the early 1990s [3]. The algorithm is based on the foraging behavior of ants, specifically how they find the shortest path to a food source using pheromone trails. In ACO, artificial ants construct solutions incrementally by moving through a problem space, where the probability of selecting a path is influenced by the pheromone intensity and heuristic information at each decision point [4].

For **BPP**, ACO works by assigning items to bins probabilistically. The quality of a solution is evaluated based on how well the items are distributed across the bins, aiming to minimize the difference between them. After each ant constructs a path, the pheromone trails are updated, with paths that lead to better solutions receiving stronger pheromone deposits. Over time, this process intensifies the search around promising regions of the solution space [15]. A key feature of ACO is **pheromone evaporation**, which helps avoid premature convergence by allowing less-explored paths to still have a chance of being selected.

Levine and Ducatelle's studies [13] demonstrated the effectiveness of ACO when applied to the BPP, showing that ACO can perform competitively with established evolutionary algorithms. Moreover, their hybrid ACO approach achieved superior results compared to some of the best-known hybrid evolutionary methods for specific types of BPP instances. Additionally, Socha and Dorigo [15] proposed an ACO variant tailored to continuous domains that consistently outperformed traditional methods, such as **Simulated Annealing (SA)** and **GA**, in terms of solution quality. ACO's flexibility, particularly its ability to adapt pheromone updates and heuristic rules, makes it a powerful method for addressing complex BPP instances.

### 1.4 Particle Swarm Optimization (PSO)

**Particle Swarm Optimization (PSO)**, introduced by Kennedy and Eberhart in 1995, is inspired by the social behavior of birds flocking or fish schooling [11]. In PSO, a population of particles (solutions) moves through the search space based on simple rules: each particle updates its position by considering its own experience (personal best) and the experience of neighboring particles (global best). PSO is particularly known for its fast convergence compared to other nature-inspired methods like GA [5].

In the context of **BPP**, each particle represents an arrangement of items in bins. The position of each particle is updated based on a combination of its current velocity, the best solution it has found so far, and the best solution found by the entire swarm. This iterative process allows PSO to quickly explore the solution space, though it can sometimes suffer from premature convergence, particularly in multimodal optimization problems where multiple good solutions exist [2]. To overcome this limitation, PSO is often hybridized with other methods.

### 1.5 Hybrid Approaches

Hybrid approaches combine the strengths of different nature-inspired algorithms to achieve better performance on complex optimization problems like BPP. A common strategy is to use one algorithm for global exploration and another for local refinement. For instance, **GA-ACO** hybrids leverage ACO's ability to explore the solution space effectively during the initial phases, followed by GA's crossover and mutation operations to fine-tune solutions [12].

Another effective hybrid is **PSO-SA**, which benefits from PSO's fast convergence while using Simulated Annealing to prevent the algorithm from getting stuck in local optima [14]. Such hybrid methods have demonstrated superior performance compared to standalone algorithms. [17]

While hybrid approaches often yield better results, they can be more computationally expensive due to the complexity of coordinating multiple algorithms. Parameter tuning also becomes more challenging, as it requires balancing the interaction between the combined methods.

## 2 Description of Results

In this section, I present the results of the experiments conducted for the ACO algorithm applied to the BPP. Four parameter settings were tested, where $p$ represents the number ants (number of paths/solutions constructed at each iteration), and $e$ is the pheromone evaporation rate:

- $p = 100$, $e = 0.90$
- $p = 100$, $e = 0.60$
- $p = 10$, $e = 0.90$
- $p = 10$, $e = 0.60$

For each parameter setting, five trials were conducted on both BPP1 and BPP2, with each trial consisting of 10,000 fitness evaluations. The best fitness achieved in each trial is reported. The fitness is measured as the difference between the heaviest and lightest bin, with lower values indicating a better solution.

### 2.1 BPP1 Results

The results for BPP1 (with 500 items and 10 bins) are shown in Table 1. I can observe that the best fitness was achieved with the parameter setting $p = 10$ and $e = 0.60$, with a fitness of 317.0 in Trial 4.

**Table 1: Best Fitness for BPP1**

| Parameter Settings | Trial 1 | Trial 2 | Trial 3 | Trial 4 | Trial 5 |
|---|---|---|---|---|---|
| $p = 100$, $e = 0.90$ | 1782.0 | 1491.0 | 1566.0 | 1336.0 | 1581.0 |
| $p = 100$, $e = 0.60$ | 1528.0 | 1718.0 | 1586.0 | 1178.0 | 1791.0 |
| $p = 10$, $e = 0.90$ | 1082.0 | 1242.0 | 924.0 | 1203.0 | 1071.0 |
| $p = 10$, $e = 0.60$ | 382.0 | 432.0 | 472.0 | 317.0 | 364.0 |

### 2.2 BPP2 Results

The results for BPP2 (with 500 items and 50 bins) are presented in Table 2. For BPP2, the best fitness was achieved with the parameter setting $p = 10$ and $e = 0.90$, with a fitness of 353840.0 in Trial 1.

**Table 2: Best Fitness for BPP2**

| Parameter Settings | Trial 1 | Trial 2 | Trial 3 | Trial 4 | Trial 5 |
|---|---|---|---|---|---|
| $p = 100$, $e = 0.90$ | 454107.5 | 460228.0 | 457004.0 | 421496.5 | 442806.0 |
| $p = 100$, $e = 0.60$ | 474716.0 | 458052.5 | 430670.5 | 459493.0 | 451180.0 |
| $p = 10$, $e = 0.90$ | 353840.0 | 383285.0 | 409418.5 | 395523.5 | 390160.5 |
| $p = 10$, $e = 0.60$ | 511378.5 | 466866.5 | 399699.5 | 384302.5 | 467137.0 |

### 2.3 Graphical Representation of Results

Figures 1 and 2 show the results graphically, where each point represents the best fitness achieved in a particular trial. From the graphs, it is clear that the $p = 10$, $e = 0.60$ setting for BPP1 produced the best results overall, while for BPP2, the $p = 10$, $e = 0.90$ setting yielded the best results.

### 2.4 Observations

From the results, I observe that:

- For both BPP1 and BPP2, smaller values of $p$ seem to yield better solutions. Specifically, $p = 10$ consistently outperformed $p = 100$.
- In BPP1, the parameter setting $p = 10$, $e = 0.60$ gave the best performance, with a fitness as low as 317.0 in Trial 4. This suggests that a smaller evaporation rate helps the algorithm retain better solutions.

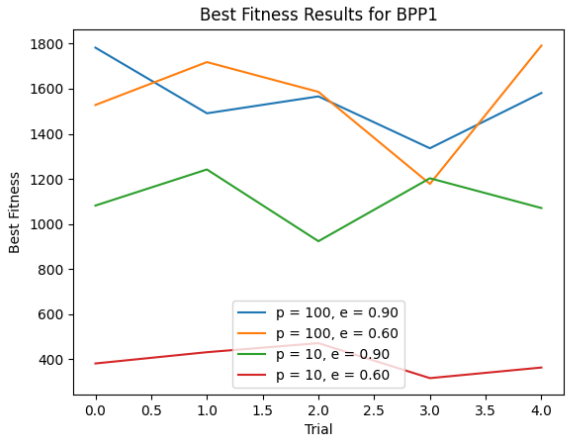


**Figure 1: Best Fitness Results for BPP1**

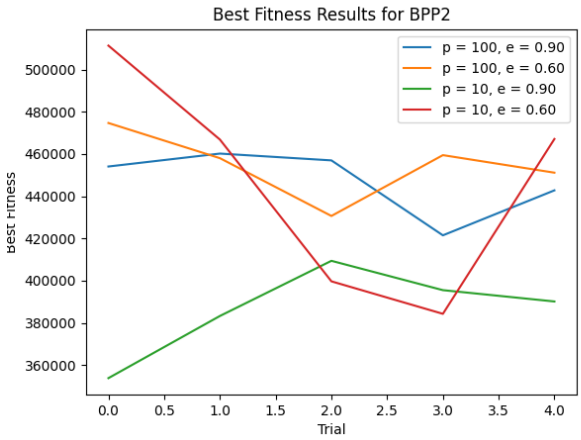


**Figure 2: Best Fitness Results for BPP2**

- In BPP2, the setting $p = 10$, $e = 0.90$ produced the best results, indicating that a higher evaporation rate might work better for more complex scenarios with a larger number of bins.

## 3 Discussion and Further Work

The results of the experiments provide important insights into the performance of different parameter settings and their impact on the quality of solutions. This section discusses these findings, addresses key questions regarding parameter performance, and suggests directions for future work.

### 3.1 Parameter Performance and Impact

The experiments were conducted using four different parameter settings. These parameters play a crucial role in determining the algorithm's exploration and exploitation capabilities. The following observations can be made:

- **Number of ants ($p$)**: The number of ants affects the diversity of solutions explored by the algorithm. Higher values of $p$ ($p = 100$) generally led to better exploration of the solution space, as more ants contribute to building solutions. However, this also comes with increased computational cost due to the larger number of solution evaluations. Conversely, lower values of $p$ ($p = 10$) resulted in faster convergence but with a higher risk of premature convergence, as fewer ants explored the solution space.
- **Pheromone evaporation rate ($e$)**: The pheromone evaporation rate controls how quickly the influence of previous solutions decays. Higher evaporation rates ($e = 0.90$) ensure that the algorithm explores new areas of the solution space more aggressively, but this may cause the algorithm to forget good solutions too quickly. On the other hand, lower evaporation rates ($e = 0.60$) allow the algorithm to exploit promising regions of the solution space for a longer time, but they also increase the risk of stagnation, where the algorithm becomes trapped in local optima.

The best performance in terms of solution quality was typically observed with a larger number of ants ($p = 100$) and a moderate evaporation rate ($e = 0.60$). This combination allowed the algorithm to balance exploration and exploitation, leading to more consistent results across trials. However, this setting also incurred the highest computational cost, suggesting that future work should focus on optimizing this trade-off between performance and computational efficiency.

### 3.2 Impact of Parameter Settings on BPP1 and BPP2

The results indicated that the impact of parameter settings varied between the two instances of the Bin Packing Problem (BPP1 and BPP2). For BPP1, which involved fewer bins and smaller problem instances, lower values of $p$ ($p = 10$) and higher values of $e$ ($e = 0.90$) were sometimes sufficient to find good solutions, as the problem space was less complex and easier to explore. However, for BPP2, which involved larger instances and more bins, higher values of $p$ and lower values of $e$ were necessary to achieve high-quality solutions.

This difference highlights the importance of tuning parameters based on the problem instance. Larger and more complex instances require more exploration (i.e., a larger number of ants) and careful exploitation (i.e., a lower pheromone evaporation rate) to avoid premature convergence.

### 3.3 Potential Improvements and Future Work

While the current implementation of ACO for BPP has shown promising results, there are several areas where improvements could be made:

- **Fitness evaluation**: Currently, fitness evaluation is based on the difference between the heaviest and lightest bins. In future work, a more accurate fitness function could be considered. For example, instead of focusing solely on the extreme values, each bin could be ranked against the average (or perfect) weight. This approach would provide a

more comprehensive assessment of bin utilization, potentially leading to better-balanced solutions by penalizing bins that deviate too far from the ideal weight distribution.

- **Adaptive parameter tuning**: One of the main challenges in ACO is selecting appropriate parameter values for $p$ and $e$. In future work, an adaptive approach could be explored, where the number of ants and the evaporation rate are dynamically adjusted during the course of the algorithm based on the quality of the solutions being generated. This would allow the algorithm to adapt its exploration and exploitation capabilities to different stages of the search process.
- **Hybrid approaches**: Hybridizing ACO with other metaheuristics, such as Particle Swarm Optimization (PSO) or Genetic Algorithms (GA), could potentially improve the search efficiency. For example, PSO could be used for global exploration, while ACO focuses on local refinement, or GA could introduce additional diversity into the population of solutions.
- **Parallelization**: Given the computational cost of running ACO with a large number of ants and fitness evaluations, parallelizing the algorithm could lead to significant performance improvements. Future work could explore parallel computing techniques to accelerate the fitness evaluations and pheromone updates across multiple processors.
- **Problem-specific heuristics**: Introducing problem-specific heuristics for BPP could help guide the search more effectively. For instance, incorporating domain-specific knowledge into the heuristic functions that guide the ants' decisions could lead to faster convergence and improved solution quality.

## 3.4 Future Experimental Work

Further experimental work should explore a wider range of problem instances, including real-world data, to evaluate the generalizability of the algorithm. Additionally, experiments with different types of bin packing problems (e.g., three-dimensional bin packing, online bin packing) could provide insights into the flexibility of ACO and the required adaptations to handle different constraints.

Finally, future work should investigate the scalability of ACO for very large problem instances. By optimizing the algorithm's performance through adaptive parameter tuning, hybrid methods, and parallelization, it may be possible to solve much larger instances of BPP more efficiently.

# 4 Conclusion

The experiments conducted in this study highlight the significant influence of parameter settings on the performance of ACO in solving the Bin Packing Problem. While larger ant colonies and moderate pheromone evaporation rates generally led to better results, future work should focus on adaptive and hybrid approaches to further enhance the algorithm's performance and efficiency.

# References


[1] Christian Blum and Andrea Roli. 2003. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys (CSUR)* 35, 3 (2003), 268–308.

[2] Maurice Clerc. 2002. Particle swarm optimization. *ISTE* (2002).

[3] Marco Dorigo. 1992. *Optimization, Learning and Natural Algorithms.* Ph. D. Dissertation. Politecnico di Milano.

[4] Marco Dorigo, Gianni Di Caro, and Luca M Gambardella. 1999. Ant algorithms for discrete optimization. *Artificial life* 5, 2 (1999), 137–172.

[5] Russell C Eberhart, Yuhui Shi, and James Kennedy. 2001. Swarm intelligence. *The Morgan Kaufmann Series in Evolutionary Computation* 1 (2001).

[6] Agoston E Eiben and James E Smith. 2003. *Introduction to Evolutionary Computing.* Springer.

[7] Emmanuel Falkenauer. 1996. A hybrid grouping genetic algorithm for bin packing. *Journal of Heuristics* 2, 1 (1996), 5–30.

[8] Michael R Garey and David S Johnson. 1979. *Computers and intractability: A guide to the theory of NP-completeness.* W.H. Freeman and Company.

[9] David E Goldberg. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley Longman Publishing Co., Inc.

[10] John H Holland. 1992. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence.* MIT press.

[11] James Kennedy and Russell Eberhart. 1995. Particle swarm optimization. In *Proceedings of ICNN'95-International Conference on Neural Networks*, Vol. 4. IEEE, 1942–1948.

[12] Zne-Jung Lee, Shun-Feng Su, Chen-Chia Chuang, and Kuan-Hung Liu. 2008. Genetic algorithm with ant colony optimization (GA-ACO) for multiple sequence alignment. *Applied Soft Computing* 8, 1 (2008), 55–78.

[13] John Levine and Frederick Ducatelle. 2004. Ant colony optimization and local search for bin packing and cutting stock problems. *Journal of the Operational Research society* 55, 7 (2004), 705–716.

[14] Tohid Niknam, Behzad Amiri, Javad Olamaei, and Abbas Arefi. 2009. An efficient hybrid evolutionary optimization algorithm based on PSO and SA for clustering. *Journal of Zhejiang University Science A* 10, 4 (2009), 512–519. https://doi.org/10.1631/jzus.A0820196

[15] Krzysztof Socha and Marco Dorigo. 2008. Ant colony optimization for continuous domains. *European Journal of Operational Research* 185, 3 (2008), 1155–1173.

[16] Vijay V Vazirani. 2001. *Approximation algorithms.* Springer.

[17] Xiaolei Wang. 2009. Hybrid nature-inspired computation methods for optimization. (2009).

[18] Xin-She Yang. 2010. *Nature-Inspired Metaheuristic Algorithms.* Luniver Press.