Bin-Packing Problem with Ant Colony Optimization

Ant Colony Optimization Algorithm Overview

Ant Colony Optimization (ACO) is a metaheuristic optimization algorithm used to find an optimal solution from a finite set of solutions. It belongs to a family of algorithms called Swarm Intelligence, which are inspired by the collective behaviour and communication within social organisms such as bees, ants, and birds, leveraging the concept of collaboration. In ant colonies, ants communicate by releasing pheromones to lead each other in specific paths; for example, an ant would release more pheromone to notify other ants of a short path that leads to food, instead of leaving them to roam other, less efficient paths in search for it.[1]

Bin-Packing Problem Overview

The bin-packing problem (BPP) involves a set number of *b* bins, and *n* items with different weights to be stored in these bins. Our goal is to distribute the weight of the items between the bins as evenly as possible.

Before we begin our analysis, let's define the meaning and anticipated impact of the number of paths *p* and evaporation rate *e* on our algorithm:

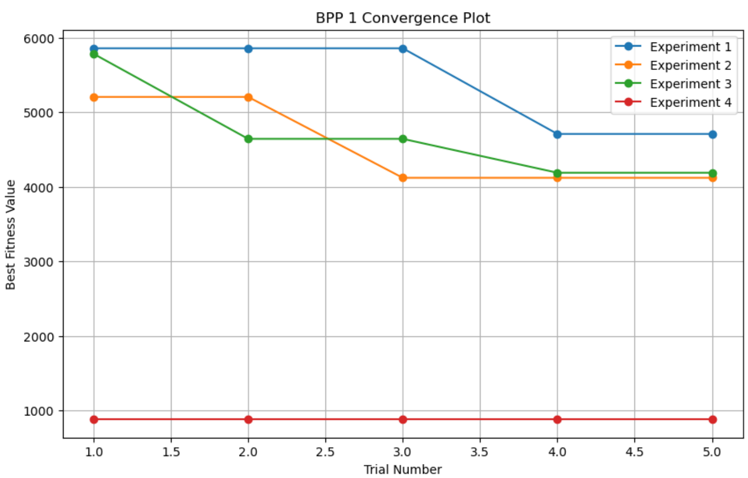
- <u>Num. paths *p*:</u> The number of paths directly affects the exploration of the search space, a larger number of paths means that the search space is being explored more thoroughly. However, there is trade-off between exploration and exploitation in this case, a larger number of paths might be able to discover better solutions, but is more computationally expensive, while a smaller number of paths exploits our existing knowledge of promising regions in the search space, and explores those only, leading to faster convergence. Although computationally cheaper, this exploitation may lead to getting stuck in a local optima.
- <u>Evaporation rate *e*:</u> The evaporation rate affects how quickly the pheromone values evaporate over time. A higher evaporation rate could lead to more dynamic exploration, as the ants aren't as heavily influenced by the previous good solutions, this means a local optima could be avoided at the likely expense of slower convergence. A lower evaporation rate means that the ants exploit their knowledge of good paths and could lead to faster convergence at the risk of getting stuck in a local optima.

Bin-Packing Problem 1: Result Analysis

Table 1: BPP1 (*b=10)* Fitness results for each trial, with different number of paths *p* and evaporation rate *e*. Fitness is measured as the difference between the weight of the largest and smallest bin.

Experiment 1 (*p=100, e=0.9)*		**Experiment 2 (*p=100, e=0.5)***		**Experiment 3 (*p=10, e=0.9)***		**Experiment 4 (*p=10, e=0.5)***	
Trial	Best Fitness	Trial	Best Fitness	Trial	Best Fitness	Trial	Best Fitness
1:	5856	1:	5204	1:	5782	1:	888
2:	5856	2:	5204	2:	4642	2:	888
3:	5856	3:	4120	3:	4642	3:	888
4:	4708	4:	4120	4:	4188	4:	888
5:	4708	5:	4120	5:	4188	5:	888
Overall Best: 4708		*Overall Best:* 4120		*Overall Best:* 4188		*Overall Best:* 888	

Figure 1: BPP1 Convergence plot for each experiment, to show the progress made by the algorithm in each of the 5 trials. The fitness value at each trial was accepted only if it improved from the last trial. If not, it was rejected.



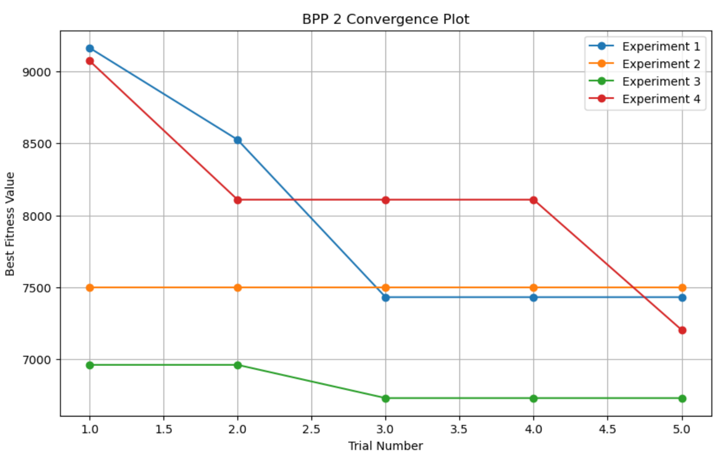
Experiment 1 converged slower, and didn't find the best fitness value relatively. This could be because of the *high* number of paths and *high* evaporation rate, meaning that the ants had to explore more paths, with less knowledge of promising paths – practically a blind search.
Experiment 3, with *few* paths and *high* evaporation rate, performed better than 1, but converged slower than Experiment 2, which had a *lower* evaporation rate. The best fitness it found was also slightly lower than Experiment 2. This indicates the importance of the pheromones in this specific problem, as they were better at leading the ants towards the best solution.
Experiment 2 converged faster than 1 because the evaporation rate was lower, meaning although it had lots of paths to explore, it had a decent idea of where the good solutions lied. The solution it found was the second best, further proving the importance of the pheromones.
Experiment 4 had the best combination of number of paths and evaporation rate, and it converged significantly faster than the others. This is likely because it took advantage of the exploration-exploitation trade-off, it didn't explore too many paths (allowing it to converge faster), and exploited the knowledge the pheromones granted it (as they weren't evaporating at a high rate).
Based on the observations we made, we can make a few inferences about our search landscape and the causes of these results. The search space for BPP1 is relatively small, it's likely not rugged and has one peak (optimal solution), which is why increasing exploration was not necessary for finding optimal solutions as the exploitation of known promising paths.

Bin-Packing Problem 2: Result Analysis

Table 2: BPP2 (*b=50)* Fitness results for each trial, with different number of paths *p* and evaporation rate *e*. Fitness is measured as the difference between the weight of the largest and smallest bin.

Experiment 1 ***(p=100, e=0.9)***		**Experiment 2** ***(p=100, e=0.5)***		**Experiment 3** ***(p=10, e=0.9)***		**Experiment 4** ***(p=10, e=0.5)***	
Trial	Best Fitness	Trial	Best Fitness	Trial	Best Fitness	Trial	Best Fitness
1:	9165	1:	7497	1:	6960	1:	9075
2:	8526	2:	7497	2:	6960	2:	8109
3:	7431	3:	7497	3:	6729	3:	8109
4:	7431	4:	7497	4:	6729	4:	8109
5:	7431	5:	7497	5:	6729	5:	7203
Overall Best: 7431		*Overall Best:* 7497		*Overall Best:* 6729		*Overall Best:* 7203	

Figure 1: BPP1 Convergence plot for each experiment, to show the progress made by the algorithm in each of the 5 trials. The fitness value at each trial was accepted only if it improved from the last trial. If not, it was rejected.



Experiment 1 had *high* exploration and *high* evaporation rate. We observe that it converged relatively well in comparison to Experiment 4, which has a *low* exploration and *low* evaporation rate. Although Experiment 4 found a better fitness, it took longer to converge. Let's observe the results of Experiment 3, which, like Experiment 1, had a high evaporation rate; we observe that it too converged relatively fast. This is likely an indication that a low evaporation rate was slowing down convergence and possibly leading the algorithm to get stuck in a local optima, as we observe in Experiment 2. This is also the reason why Experiment 4 took relatively longer to converge, even if it ended up finding a slightly better solution in comparison to Experiment 1.

Overall, Experiment 3 with a small number of paths and high evaporation rate performed the best, as it had the best balance of the exploration-exploitation trade off.

Based on the observations we made, we could infer that the search landscape for BPP2 was more rugged than BPP1.

It is likely that Experiments 3 and 4 were able to find better solutions (with path set to 10) as they were taking bigger steps throughout the search space, which is good in a rugged search space as we're not exploiting all potential promising regions and getting stuck in local optima. This could explain why although Experiment 4 started with bad fitness, it managed to converge towards significantly better solutions and rank second best overall.

Potential Alternative Algorithm

One algorithm that could potentially work better on our bin-packing problem could be teh Particle Swarm Optimization (PSO) algorithm which, like Ant Colony Optimisation, also belongs to a family of Swarm Intelligence algorithms.

PSO mimics the behaviour of swarms of fish and birds, where each particle has an associated position, velocity, and fitness value. Each particle also keeps track of the particle in position with the best fitness and the associated fitness value, the particle adjust their own position based on the globally best fit position and value. (https://www.geeksforgeeks.org/particle-swarm-optimization-pso-an-overview/)

PSO is particularly suitable for optimization problems with continuous search spaces, where solutions are represented as particles moving through the search space. In the bin packing problem, the continuous nature of the search space can be represented by the placement of items in bins, and the movement of particles can correspond to rearranging the items among bins.

It can continuously adjust the placement of items in bins, allowing for finer adjustments and the potential to escape local optima. It also achieves a balance between exploration and exploitation by adjusting particle positions based on personal and neighborhood information.

References

1. Introduction to Ant Colony Optimization. GeeksforGeeks. Published May 15, 2020. https://www.geeksforgeeks.org/introduction-to-ant-colony-optimization/
2. Particle Swarm Optimization (PSO) - An Overview. GeeksforGeeks. Published April 22, 2021. https://www.geeksforgeeks.org/particle-swarm-optimization-pso-an-overview/