Ant Colony Optimization, and other Nature Inspired solutions to the Bin Packing Problem

Candidate Number: 720052156

Abstract

In this paper, natural solutions to the bin-packing problem will be examined, and one particular version of each will be explored and evaluated.

1 Literature Review

The bin packing problem is a mathematical problem wherein n objects with varying weights must be assigned to bins with a set capacity, such that each item is placed in a bin, no bins exceed capacity, and the minimum number of bins are used. This problem is an NP-complete one, meaning that there exist no algorithms that reliably provide perfect solutions, leading to several algorithms, each producing "good enough" solutions in polynomial time [3]. The bin packing problem has multiple permutations varying in factors such as dimensionality, whether the bin's capacities are finite, and the desired outcome [6]. Being an NP-complete problem, each potential algorithmic solution could have the potential to be better than the last, and since the problem is so analogous to real-life situations [9], solutions could come from anywhere.

The immediate most simple solutions are heuristics; universal simple rules than can be applied to reach a decent, deterministic outcome quickly. For some situations, this is adequate, but for many, better solutions are a necessity, or worth the extra computational and time cost . These better solutions can be found using meta-heuristic algorithms, which not only find solutions, but compare and iterate on them to optimize the result. A great many of these meta-heuristic algorithms, a sizeable majority in some cases, are inspired by the oldest tried and true problem solver in history: nature.

Nature-inspired mathematical algorithms are something of a doctrine of their own, as old as mathematics itself, but increasing in popularity in recent decades due to its proficiency in providing clear metaphors to base processes on [5]. As well as this, natural entities such as plants, animals and fungi, exhibit both effective methods of solving short-term problems of varying scale but also adapt to more all-encompassing problems with little or no required effort or intelligence, guided by the process of natural selection. Both on a micro and macro scale, there are many analogies to pull from. Sometimes, however, it's pertinent to bear in mind the limitations of natural problem-solving. Without sapience or foresight, nature can reach solutions which, while functional, are far from optimal, and it's always worth considering whether a natural solution is founded on its own merits, or for its own sake [8]. Perhaps as a result of this vitally important filter, viable nature-inspired algorithms may not be many but make up for it with their efficacy. For the remainder of the sections, some of these algorithms will be explained, analysed, and compared.

1.1 Genetic Algorithms

Genetic algorithms (GAs) are a staple of nature-inspired computing, both for their effectiveness and broadness of application [2]. They operate by creating an initial population of solutions, and via some genetic function involving mutation, crossover or both, fine tuning towards some optimized solution. This paper [13] demonstrates the effectiveness of such algorithms on a 1-dimensional bin packing problem (1BP) similar to the one that will be examined in upcoming sections of this paper. The algorithm examined in the paper shows how, with very little theoretical complexity, very effective solutions can be reached very efficiently and reliably using this method. Particularly shown in this paper, how a problem is encoded and represented before the algorithm is in use is vitally important, as genetic algorithms are so conceptual that great consideration has to be paid to which aspects of solutions should vary, and by which means. As mentioned in the conclusion of the paper, this was a blind spot in GA studies, as it was previously thought that GAs were inherently unsuited to grouping problems [7].

1.2 Firefly Algorithms

Firefly algorithms are a type of meta-heuristic algorithm which, similarly to GAs, fine-tune solutions by combining them with each other. They distinguish themselves via the means of selection, mimicking fireflies' mating patterns, each solution occupying some simulated space, and radiating a brightness representing their fitness according to their objective function. Fireflies are attracted to the light which has some ideal combination of proximity and brightness, toward which they move to eventually combine with via some genetic function [15]. The application of a firefly algorithm on a 1BP can be seen in this paper [17], and can be seen to perform reasonably well in hard problems compared to First Fit and Best Fit heuristics, but, as with genetic algorithms, care needs to be put into fitting the algorithm to the particular problem, and fine-tuning the parameters.

1.3 Cuckoo Search Algorithm

Cuckoo search algorithms are a meta-heuristic algorithm of two parts: nests, containing eggs representing potential solutions and cuckoos representing agents which choose nests biased by the nest's fitness. In each generation, cuckoos fly according to a Levy flight pattern[4] and choose nests to lay their eggs in, prioritising nests other than their own. Nests with more eggs maintain prevalence in the following generations, and in inhabited nest each nest, one egg will hatch, yielding a cuckoo [16]. One of the primary advantages of this algorithm, beyond its propensity for good results, is its low number of parameters, allowing it to easily be fine-tuned to achieve optimized outcomes. According to this paper [18], this algorithm both benefits and suffers from the Levy flights, producing high quality results, but doing so very slowly due to the inherent randomness. To temper this without losing the explorative nature which is a huge strength, a Ranked Order Value rule maps a continuous state space to a discrete one, speeding up convergence time without significantly hindering quality of results. This can be particularly effective, not only outperforming heuristics, but indeed some meta-heuristics as well, namely the Ant System algorithm, which we will be analysing for the remaining sections in this paper.

1.4 Whale Optimization Algorithm

Whale optimization is an algorithm which simulates the hunting behaviour of humpback whales. Agents in this algorithm start off randomly distributed, then spiral around the fittest agent [10]. As an optimisation method, this is exactly what it needs to be, which is to say simple, effective, and broad in its application. Shown in this paper [1] for the 1BP, it can be adapted to particularly effective meta-heuristic, dubbed Improved Levy-Based Whale Optimisation Algorithm (ILWOA). As the name suggests, its uses a similar searching behaviour as Cuckoo Search algorithms, using Levy flight patterns. This is particularly suited to the problem owing to its ability to hone in on solutions as clusters of agents who converge independently, enabling for multiple very distinct avenues to be investigated in one run of the algorithm.

1.5 Comparison

All of these meta-heuristics can provide varying degrees of success for varying levels of sacrifice, but they still differ in relevant ways. Consideration should be paid to which technique, be it heuristic or meta-heuristic, is the most pertinent to use such that the minimum complexity (in terms of time, implementation and computation) can be employed while still receiving an ideal outcome [12]. In terms of solutions to the 1BP (accounting for the caveat of subjectivity that is implicit when an algorithms are adapted to a problem in a novel way), the hierarchy of effectiveness of each algorithm is the same as the order in which they are listed. This paper [11] offers an excellent summary of the state of nature inspired algorithms applied to the 1BP as of 2021, which has both proved to be an enormous contribution to this report, but also points out some insightful distinctions between them. The Adapted Cuckoo Search (ACS) algorithm outclasses both the ant colony optimisation seen in the following sections and the Firefly Algorithm seen earlier, owing to its effective harnessing of noisy exploration, and its few parameters lead to relatively easy implementation, making it a strong contender. ACS however, despite its adaptation to converge quicker, still seems to fall slightly short of the ILWOA seen earlier in terms of speed, and their results are typically either the same or slightly better in the ILWOA. All this said however, and important thing to consider as an effective razor, mentioned in the paper, is the no-free-lunch theory [14], which essentially posits that between two algorithms, any upper hand one may have over another in a certain field, the other will compensate when applied to another. This axiom, given the variety of presentations and permutations of the bin packing problem, urges that one should still give some thought to the matter.

Table 1: Problem 1 results

Experiment #	Minima	Average	Range
1	1336, 1832, 1655, 1330, 1162	1463.0	670
2	1866, 1448, 1702, 1307, 1509	1566.4	559
3	861, 1024, 1190, 981, 993	1009.8	329
4	381, 589, 552, 349, 376	449.4	240

2 Ant Colony Optimisation Experiment and Results

The particular bin packing problems looked at in this paper had the following specifications

Problem 1:

- 10 bins of infinite capacity
- 500 items i with weights i
- 10000 total ants

Problem 2:

- 50 bins of infinite capacity
- 500 items i with weights calculated using the following formula:
$$\frac{i^2}{2}$$
- 10000 total ants

Upon these problems, four ant colony optimisation solutions were simulated, each varying in two ways: army size and evaporation rate. The former represents how often the pheromones get updated, and the latter represents how much of an impact older pheromones have on decision making. The following inputs were used:

- Experiment 1:
  - Army Size: 100
  - Evaporation Rate: 0.9
- Experiment 2:
  - Army Size: 100
  - Evaporation Rate: 0.6
- Experiment 3:
  - Army Size: 10
  - Evaporation Rate: 0.9
- Experiment 4:
  - Army Size: 10
  - Evaporation Rate: 0.6

Fitness is measured by the difference between the volume of the fullest bin and that of the least full bin. The smallest difference is a minimum. Henceforth, a lower fitness level would be considered better. The resulting minima of each of these experiments on can be seen in Table 1 and Table 2 respectively.

The averages and ranges can be seen in Figure 3, and from these we can see the general fitness of these parameters in the averages, but also the reliability of this quality using the ranges. These are both pertinent metrics because they will govern either whether a set of parameters should be used or how many times.

The results in Tables 1 and 2 show the requested data, however, I thought it pertinent to take graphs of the minimum of each army, to get an idea of the gradient of each graph, and whether or not

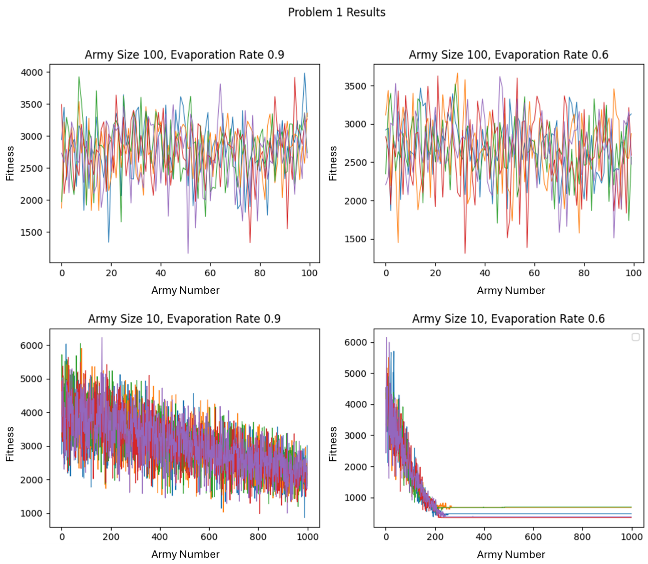

Figure 1: The Fitness curves for each experiment in problem 1

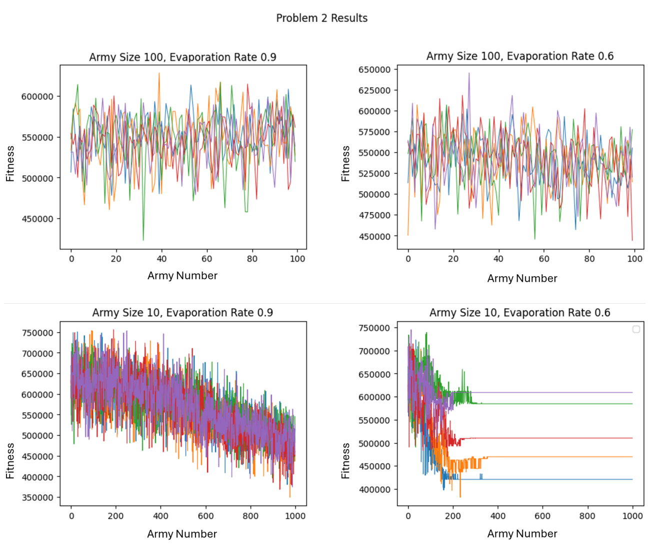

Figure 2: The Fitness curves for each experiment in problem 2

they converge. The results for problems one and two can be seen in Figure 1 and Figure 2 respectively. These give a much deeper insight into the behaviour of the ants within the algorithm, showing the effect of the independent parameters on the gradients of the curve.

One notable thing is that the seemingly total randomness of experiment 1 in problem 1 actually seemed to improve the average minimum result, compared to experiment 2. This can be attributed to its greater range and the fact that neither experiment gets so much as close to convergence within the time frame.More to the point, it shows how convergence happens in experiment four, and how wildly differently that affects the outcome. This shows the difference in the nature of the problems, and indeed that the second problem is (as could be expected by its fivefold additional graph complexity) far more risky to converge early on. This explains why the converged values are so wildly different, and typically inferior to the other curves despite the fact that they do not converge.

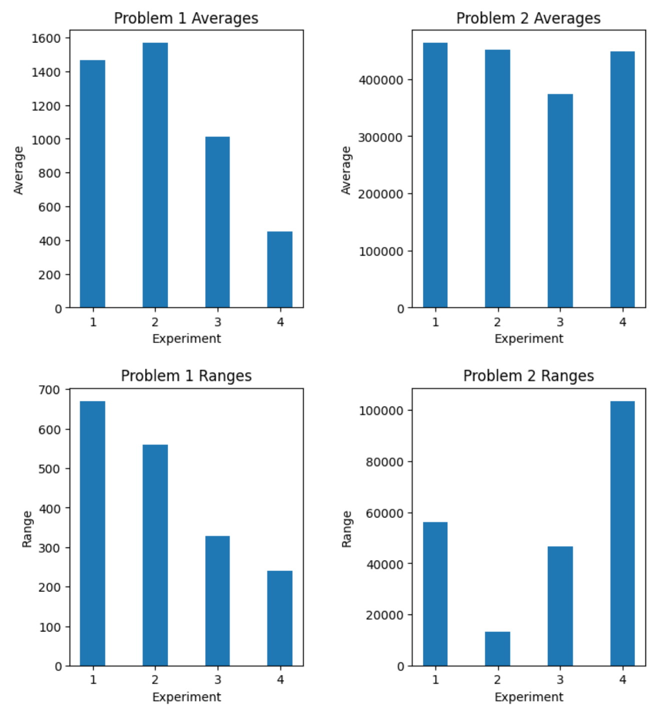

Figure 3: Ranges and averages of the results in tables 1 & 2

3 Discussion and Further Work

3.1 Q1: Best Combination of Parameters

For the first problem, the best combination of parameters within the time frame provided by the specification was 10 ants per army and 0.6 evaporation rate. This was the best because it was the only one which properly converged, and it did so at a value lower than all others. For the second problem, however, the early convergence was a hindrance, and the results were better with the same army size, but the higher pheromone evaporation rate of 0.9. That said, when the number of iterations was increased beyond the scope of the specification, the variables which led to slower convergence converged to generally better results. This, however, is with the notable exception of experiment one, which seems to be virtually indistinguishable from noise. In both problems, 100 ant armies and evaporation rate of 0.6 provided the best converged result, as can be see in Figure 4 and Figure 5. To get this to yield however, 140000 iterations were needed, which is so vastly beyond the scope that one could argue the problem's nature changes in respect to time budget, and other considerations ought to be made. I considered experimenting further to be out of scope for this coursework, considering the time cost. Even within this extended time frame however, one thing I considered worth noting was that the 100 ant army and

Table 2: Problem 2 results

Experiment #	Minima	Average	Range
1	478209, 460577, 422927, 473430, 478969	462822.4	56042
2	457113, 450521, 445669, 444285, 457654	451048.4	13369
3	379976, 349256, 395867, 368372, 374277	373549.6	46611
4	397559, 381836, 490529, 469943, 501060	448185.4	103501

0.6 evaporation rate still seemed indistinguishable from noise. I speculate that it may converge if given a long enough time period, perhaps to an even better solution, but for a problem like this, I believe that this would be a colossal waste of time and resources.

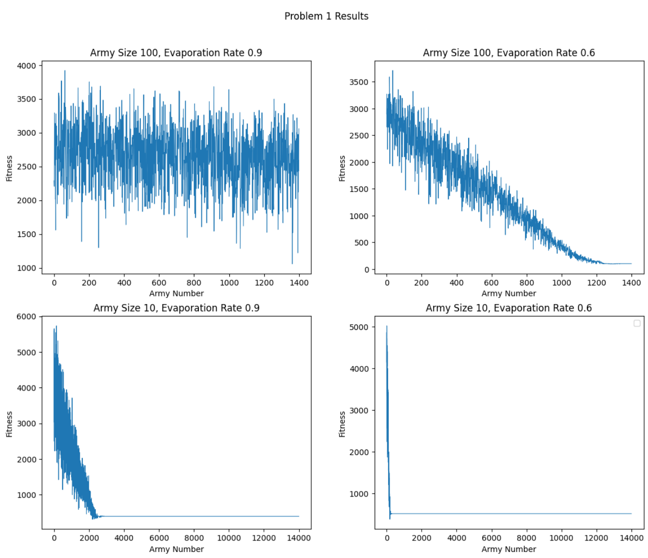

Figure 4: The convergence shapes of Problem 1 after 140000 iterations

3.2 Q2: Explain your answer to Q1

I believe my results to be as they are, in short, because a more frequently and potently updated pheromone table allows ants to more quickly and consistently take better paths as more generations come.

3.3 Q3: How Parameters Affect Outcome

Both small army sizes and low evaporation rates correlate directly with more exploitation, but less exploration. In this particular scenario, the balance between the two needs to lean more toward this due to the amount of runtime considered reasonable. As mentioned before however, the greater bias toward exploitation leads to inferior convergence when a longer time scale is applied.

3.4 Q4: Potentially Better Algorithms

From my research into similar solutions to similar problems, I'd speculate that ACS or ILWOA would have been superior solutions to this problem, owing to their faster convergence time and, in the

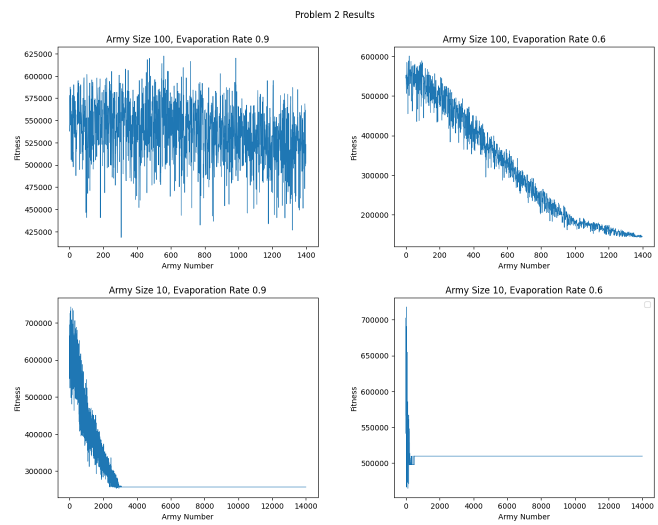

Figure 5: The convergence shapes of Problem 2 after 140000 iterations

case of ACS, less factors to consider. Indeed, any swarm intelligence algorithm that is able to traverse the state-space in a way that is more intelligent than what essentially reduces to lightly guided trial and error is likely to converge quicker, and with the adaptations in this regard made to ACS and ILWOA I'm lead to assume they would be far superior.

References

[1] Mohamed Abdel-Basset, Gunasekaran Manogaran, Laila Abdel-Fatah, and Seyedali Mirjalili. 2018. An improved nature inspired meta-heuristic algorithm for 1-D bin packing problems. *Personal and Ubiquitous Computing* 22, 5-6 (10 2018), 1117–1132. https://uoelibrary.idm.oclc.org/login?url=https://www.proquest.com/scholarly-journals/improved-nature-inspired-meta-heuristic-algorithm/docview/2017323297/se-2 Copyright - Personal and Ubiquitous Computing is a copyright of Springer, (2018). All Rights Reserved; Last updated - 2023-12-03.

[2] AnalytixLabs. 2024. A Complete Guide to Genetic Algorithm — Advantages, Limitations More. *Medium* (1 2024). https://medium.com/@byanalytixlabs/a-complete-guide-to-genetic-algorithm-advantages-limitations-more-738e87427dbb

[3] Joan Boyar and Lene M. Favrholdt. 2007. The relative worst order ratio for online algorithms. *ACM Trans. Algorithms* 3, 2 (May 2007), 22–es. https://doi.org/10.1145/1240233.1240245

[4] Mridul Chawla and Manoj Duhan. 2018. Levy Flights in Metaheuristics Optimization Algorithms – A Review. *Applied Artificial Intelligence* 32, 9-10 (2018), 802–821. https://doi.org/10.1080/08839514.2018.1508807 arXiv:https://doi.org/10.1080/08839514.2018.1508807

[5] Raymond Chiong. 2009. *Nature-Inspired Algorithms for Optimisation*. Springer Berlin, Heidelberg, p. VII. https://doi.org/10.1007/978-3-642-00267-0

[6] Henrik I. Christensen, Arindam Khan, Sebastian Pokutta, and Prasad Tetali. 2017. Approximation and online algorithms for multidimensional bin packing: A survey. *Computer Science Review* 24 (2017), 63–79. https://doi.org/10.1016/j.

cosrev.2016.12.001

[7] Emanuel Falkenauer. 1994. A New Representation and Operators for Genetic Algorithms Applied to Grouping Problems. *Evolutionary Computation* 2, 2 (06 1994), 123–144. https://doi.org/10.1162/evco.1994.2.2.123 arXiv:https://direct.mit.edu/evco/article-pdf/2/2/123/1492760/evco.1994.2.2.123.pdf

[8] Stephen Freeland. 2003. Three Fundamentals of the Biological Genetic Algorithm: A Biologist's View of Biological Inspiration vs. Biology Envy. *Genetic programming theory and practice* (2003), 303–311.

[9] Joseph Malkevitch. 2004. *Bin Packing and Machine Scheduling*. https://www.ams.org/publicoutreach/feature-column/fcarc-packings3

[10] Seyedali Mirjalili and Andrew Lewis. 2016. The Whale Optimization Algorithm. *Advances in Engineering Software* 95 (2016), 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008

[11] Chanaleä Munien and Absalom E. Ezugwu. 2021. Metaheuristic algorithms for one-dimensional bin-packing problems: A survey of recent advances and applications. *Journal of Intelligent Systems* 30, 1 (2021), 636–663. https://doi.org/doi:10.1515/jisys-2020-0117

[12] R. Russel Rhinehart. 2019. Optimization algorithm selection for process applications. *InTech* (11-12 2019). https://www.isa.org/intech-home/2019/november-december/features/optimization-algorithm-selection-for-process-appli

[13] Adam Stawowy. 2008. Evolutionary based heuristic for bin packing problem. *Computers Industrial Engineering* 55, 2 (2008), 465–474. https://doi.org/10.1016/j.cie.2008.01.007

[14] D.H. Wolpert and W.G. Macready. 1997. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation* 1, 1 (1997), 67–82. https://doi.org/10.1109/4235.585893

[15] Xin-She Yang. 2009. Firefly Algorithms for Multimodal Optimization. In *Stochastic Algorithms: Foundations and Applications*, Osamu Watanabe and Thomas Zeugmann (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 169–178.

[16] Xin-She Yang and Suash Deb. 2009. Cuckoo Search via Lévy flights. In *2009 World Congress on Nature Biologically Inspired Computing (NaBIC)*. 210–214. https://doi.org/10.1109/NABIC.2009.5393690

[17] R. Yesodha and T. Amudha. 2013. Effectiveness of FireFly Algorithm in solving Bin Packing Problem. *International Journal of Advanced Research in Computer Science and Software Engineering* 3, 5 (5 2013).

[18] Zendaoui Zakaria and Abdesslem Layeb. 2016. *Adaptive Cuckoo Search Algorithm for the Bin Packing Problem*. 107–120. https://doi.org/10.1007/978-3-319-33410-3_8