ECM3412 Coursework Assessment

690019495

December 15, 2021

Abstract

The Bin-Packing Problem (BPP) is an NP-hard combinatorial optimization problem [2,4]. Its main optimization problem is pack a set of items of known sizes into a minimum number of bins [4]. Due to the complexity for large problems, it is impractical to solve for optimal solutions, hence computational approximations are better suited [3,4]. This report focuses on using an Ant Colony Optimisation as a heuristic approach to the BPP and finding the approximate best solutions for the specified problems.

1 Introduction

The task for this assignment is the implementation of the ant colony optimisation (ACO) algorithm to the problem of bin packing. The packing problem statements are:
BBP1 = 500 items of weights of i, where i equal to the item number (eg item 2 = weight 2). These items are to be packed into 10 bins.
BPP2 = 500 items of weights of i^2, where i equal to the item number ^{2} (eg item 2 = weight $2^2 = 4$). These items are to be packed into 50 bins.

The construction graph is made of values that correspond to which item should be placed in what bin. The better the bin placement, then the higher the relation value between the item and bin. Therefore, each item (1 to 500) will have b bin (dependant on which BBP problem) relationships. Example:

```
>>> Item 1:  [Bin 1 = 0.9346743310111895, Bin 2 = 0.9554999913151847,
Bin 3 = 0.13090693383092977, ....  , Bin b = n ]
>>> Item 2:  [Bin 1 = 0.8199642705160568, Bin 2 = 0.8201860263276137,
Bin 3 = 0.6032482839439789, ....  , Bin b = n ]
```

Implementation of the ACO:
1. Randomly distribute small amounts of pheromone (between 0 and 1) on the construction graph.
2. Generate a set of p ant paths from S to E.
3. Update the pheromone in your pheromone table for each ant's path according to its fitness.
4. Evaporate the pheromone for all links in the graph.
5. If a termination criterion has been reached, then stop. Otherwise return to step 2.

2 Experiments

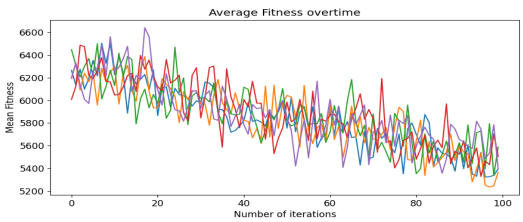
2.1 BBP1

Experiment 1: ACO with p = 100 and e = 0.9				
Run	End optimal	Maximum	Minimum	Average
1	3215	13072	1700	6228
2	2883	13912	1685	6185
3	2868	14734	1856	6185
4	2959	13776	1701	6188
5	2797	14223	1701	6223

Experiment 2: ACO with p = 100 and e = 0.5				
Run	End optimal	Maximum	Minimum	Average
1	1924	12863	1347	5858
2	1821	12926	1510	5854
3	2752	13161	1478	5852
4	2067	13042	1468	5834
5	2584	12452	1287	5839

Experiment 3: ACO with p = 10 and e = 0.9				
Run	End optimal	Maximum	Minimum	Average
1	2328	12163	1263	4892
2	1353	11921	1065	4883
3	1946	12932	1236	4832
4	1981	12265	1236	4898
5	2887	12265	911	4906

Experiment 4: ACO with p = 10 and e = 0.5				
Run	End optimal	Maximum	Minimum	Average
1	598	10760	447	935
2	432	12579	432	798
3	332	10435	318	751
4	544	11580	438	890
5	1250	11389	660	1479



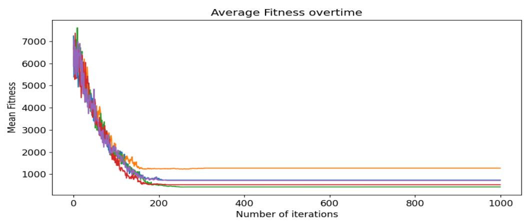


Figure 1: Left: Gradual decrease of average fitness (p=100 and e = 0.5) . Right: Fast decrease of average fitness (p = 10 and e = 0.5)

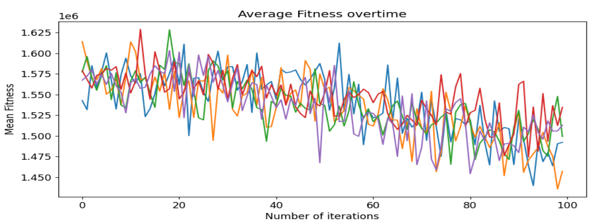
2.2 BBP2

Experiment 1: ACO with p = 100 and e = 0.9				
Run	End optimal	Maximum	Minimum	Average
1	1178984	2644599	789314	1427960
2	865473	2795415	782551	1415691
3	1010910	2735258	755646	1427058
4	901037	2757686	663458	1399390
5	1013694	2791781	742971	1401382

Experiment 2: ACO with p = 100 and e = 0.5				
Run	End optimal	Maximum	Minimum	Average
1	1007232	2684562	833685	1550642
2	1032628	2704973	828923	1537864
3	1000476	2691703	773924	1532813
4	1050975	2979666	883503	1534274
5	952498	3004887	849118	1534408

Experiment 3: ACO with p = 10 and e = 0.9				
Run	End optimal	Maximum	Minimum	Average
1	1086448	2671927	914827	1566399
2	1048650	2930265	971660	1567610
3	1114548	2709194	805372	1568508
4	1148341	2834811	926194	1569347
5	1049968	2776645	913805	1572175

Experiment 4: ACO with p = 10 and e = 0.5				
Run	End optimal	Maximum	Minimum	Average
1	912909	2304742	707186	959572
2	1276477	2519358	921816	1293638
3	1273460	2615572	1033457	1297659
4	1266798	2471455	945434	1283413
5	911842	2297469	889445	968812



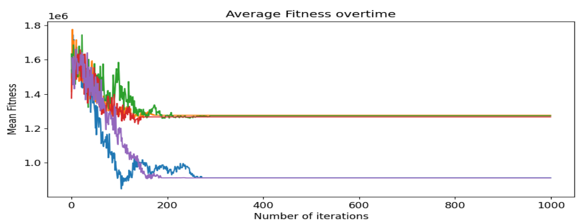


Figure 2: Left: Sporadic decrease of average fitness (p=100 and e=0.9). Right: Local optima found due to the lower number of ants not able to cover all the graph (p=10 and e=0.5)

3 Results

1) For BBP1, the best combination was p = 10 and e = 0.5. For BBP2, the best combination was p = 100 and e = 0.9

2) For BBP1, combination was best as the problem format is suited to smaller ant sizes. For p = 100, when 100 ant paths are generated, the ants traverse the graph and mostly overwrite previous best paths and worse paths. If 20 ants traverse a non ideal fitness path, then the fitness update (100/fitnessofpath) is updated each ant turn. At the end of every ants run the environment is evaporated. With e = 0.9, this is not enough to reduce the worse fitness paths. e = 0.5 is an improvement, however there are not enough fitness evaluation iterations to produce the best results. In Figure 1, it is clear that p = 100 and e = 0.5 is slowly decreasing, yet p = 10 and e = 0.5 converges on a solution much faster.

For p = 10, the small ant traversal size is less likely to scramble legitimate paths. With the evaporation value as 0.5, the paths less ideal will be reduced significantly more than e = 0.9. BBP1 is more sensitive to the fitness at the item weights are small compared to BBP2. e = 0.5 heavily removes non ideal paths, and is better for the small iteration size as this converges faster on the best approximate solutions out of the other experiments. Experiment 4 falls into a local optima. In the further analysis, it is seen that better fitness solutions exist.

For BBP2, the combination was best as the larger number of bins allowed the ant size to iterate more effectively through the graph. Due to the large size of the item weights, using e = 0.9, the fitness update was not at volatile as e = 0.5. This incremental difference worked better when the fitness evaluation was performed for the path, and the evaporation. Using a lower e value and smaller ant size will "kill" most paths, as these are not traversed often enough and the evaporation reduces the pheromone to such a small amount. Using p = 10 and e = 0.5 produced a lower average fitness, however this is false due to the paths being the same each iteration due to the graph having "dead" paths. The pheromone values are so small for possible paths that the same path is therefore chosen as this would be the highest weighted in pheromone value.

3) The parameter settings influence: The evaporation value multiplies all the pheromone weights in the graph. The lower the value, the more worse paths/not traversed paths are reduced, and the faster the algorithm converges. The higher value creates a smoother search landscape that results in a slower rate of convergence. Initially, if the ants find a good path then this will be traversed more. Using a lower 'e' value a path that was not initially found will be reduced heavily and therefore not chosen as often. This path is then reduced more every iteration. Because the paths that are not traversed as much are reduced more often (e = 0.5 is half the pheromone value every evaporating step), this can lead to a local optima seen in Figure 2.

The evaporation value is therefore dependant on the search space. For a smaller bin number and small termination criteria (10000 fitness evaluations), a small evaporation value is ideal. This value will reduce the less ideal paths more each iteration leaving less room for bad path fitness. For a larger bin number, a larger evaporation value is ideal. **Therefore, the evaporation value influences the convergence performance rate.**

The ant size allows greater exploration of the environment. For larger environments, a larger ant group can traverse more paths. This will ensure that most paths are traversed, and more likely to find better paths. If the ant group is not large enough, then pheromones on paths not traversed as often will be evaporated, decreasing the pheromone weight even more (especially if 'e' is low). This results in a local optima, as ants will most likely traverse a single path because the pheromone have been reduced to so much due to not being searched as often. The algorithm falls into the local optima discussed above. The pheromone for other paths are so reduced that these are never chosen, resulting in ants only taking the single path for the remainder of the iterations. **Therefore, the ant size influences the graph coverage which produces better fitness results and no local optima.**

4) The traditional ACO heuristic function is generated by the fitness probability of the available paths from each item. This has a great uncertainty and takes a long time to compute [2]. Even using optimal parameter settings it is not easy to search for the global optimal solution. Chen et al. [1] adopts the uses of the A* valuation in the heuristic function to let the ants drive to the point on the line between the starting point and the ending point with a greater probability [1]. This method addresses the problems of the basic ACO in initial searching for path planning applications.

This paper adds the idea of A* algorithm valuation function to the heuristic function of ACO for improving the convergence speed of the algorithm and obtaining more optimal planning paths [1]. The A* algorithm has a faster merit-seeking speed because of its unique valuation function, where the A* path finding will preferentially move towards the endpoint each time [1]. Hence, as well as the added fitness value, the local heuristic parameter is also taken into account [1]. The improved algorithm in the paper enables the Ant to find an optimal path with shorter path length in less time and fewer iterations, proving the effectiveness and adaptability of the method and therefore would be better suited to this report [1].

4 Further analysis

For each problem I will use more optimal parameters for each problem. For BBP1, using p = 10 and e = 0.5 fell into a local optima. However because the item size was small, this was not a problem as this produced the best result. Therefore, I will increase the fitness evaluations to 50000 for further range.

BPP1 optimise: ACO with p = 50 and e = 0.6				
Run	End optimal	Maximum	Minimum	Average
1	131	13441	103	2332
2	125	12077	125	2322
3	253	13338	194	2446
4	191	12069	145	2342
5	184	11787	137	2395

BBP2 optimise : ACO with p = 75 and e = 0.68				
Run	End optimal	Maximum	Minimum	Average
1	765707	2913912	622932	1347041
2	775434	3008259	624503	1341119
3	749791	13188	602762	1331276
4	788946	3101590	651659	1334163
5	746960	3057205	621949	1323426

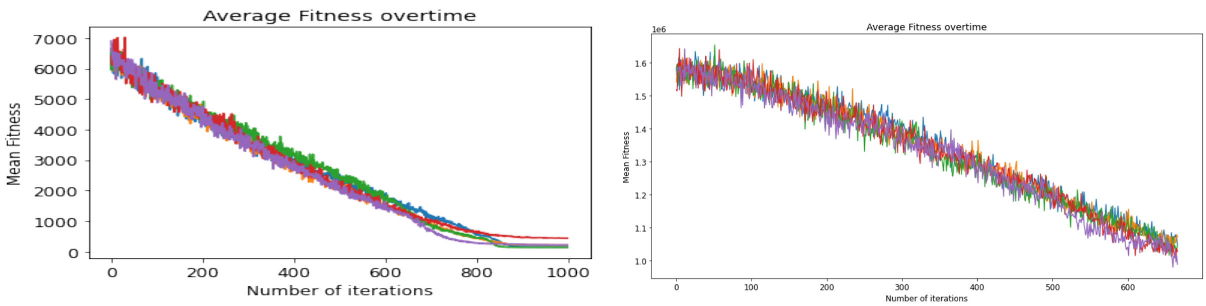


Figure 3: BBP1 Left: Near global optima is found instead of the local optima (p=60 and e = 0.6). BBP2 Right: Gradual decrease of fitness (p = 75 and e = 0.68)

From Figure 3, the left graph shows convergence less quickly. Given more fitness valuations, using a higher p value results in more graph coverage and a higher e value results in a slower convergence rate. Using e = 0.9 does not reduce unwanted paths enough that the are less likely to be chosen.

For BBP1, the number of bins is small. Therefore Experiment 4 was best due to the convergence speed. p = 50 and e = 0.6 is more optimal with 50000 iterations as there are enough ants to traverse most of the graph without overpopulation, and the evaporation rate removes unwanted paths while not falling into the local optima.

For BBP2 there is the potential for big differences between maximum and minimum bins weights. Experiment 1 was best due to graph coverage. Upon updating the pheromone values the increments cause minuscule differences to the pheromone graph. This results in a slower rate of convergence. This is demonstrated in figure 2, where p = 10 and e = 0.5 prematurely converges into the local optima. Using p = 75 and e = 0.68 allows greater coverage of the search space, while the pheromone update values are reduced more.

In conclusion, a higher evaporation rate results in a slower rate of convergence. At each bin item point, the pheromone values of the next node are similarly weighted, giving an equal the selection of each next node. The ants traverse more of the graph and is therefore more likely to approximate the optimal solution. However, converges slower than using a lower evaporation rate. This is represented best in Figure 3 where e = 0.6 as we can see a smoother convergence than the sharp alternative of e = 0.5 shown in Figure 2. If the amount of termination iterations is smaller, then the use of a lower evaporation value should be used because the algorithm can not afford to waste searching with less suitable paths. By allowing more fitness evaluations using a higher evaporation rate produces small incremental decreases of the fitness, overall producing better results.

Overall, the search space for ACO needs to be taken into consideration and the speed of which a result needs to be found. For longer searches, a higher ant size and evaporation value is better as this allows a larger coverage of the search space, thus a slower convergence but the higher possibility for a better approximate solution. For shorter searches, a fast yet good solution is achieved by using a lower evaporation rate and not as big ant size to overpopulate.

References

[1] Chen, L., Su, Y., Zhang, D., Leng, Z., Qi, Y. and Jiang, K., 2021, May. Research on path planning for mobile robots based on improved ACO. In 2021 36th Youth Academic Annual Conference of Chinese Association of Automation (YAC) (pp. 379-383). IEEE.

[2] SMIMITE, O. and AFDEL, K., Hybrid Solution for Container Placement and Load Balancing based on ACO and Bin Packing.

[3] Silva-Gálvez, A., Lara-Cárdenas, E., Amaya, I., Cruz-Duarte, J.M. and Ortiz-Bayliss, J.C., 2020, June. A Preliminary Study on Score-Based Hyper-heuristics for Solving the Bin Packing Problem. In Mexican Conference on Pattern Recognition (pp. 318-327). Springer, Cham.

[4] El-Ashmawi, W.H. and Abd Elminaam, D.S., 2019. A modified squirrel search algorithm based on improved best fit heuristic and operator strategy for bin packing problem. Applied Soft Computing, 82, p.105565.