Ant Colony Optimisation: An Approach to 0-1 Knapsack Problem

Candidate number: 018927

1 Literature Review

Our money bag problem can be described as a classic case of 0-1 Knapsack problem (KP01) problem, where each bag can be accessed once or not at all: we are given a set of 100 items, each item has an value v_j and an weight w_j. The problem is to choose a subset of the items such that the overall value is maximised, while the overall weight does not exceed the given capacity 295. Our problem can be written as follows:

$$\text{maximise} \quad \sum_{j=1}^{100} v_j x_j \tag{1}$$

$$\text{subject to} \quad \sum_{j=1}^{100} w_j x_j \leq 295, \tag{2}$$

$$x_j \in \{0, 1\},$$

$$v_j, w_j > 0, \quad j \in \{1, \ldots, 100\}.$$

The aim of our Ant Colony Optimisation algorithms is to maximise our fitness function (1).

Researches suggest KP01 are weakly NP-hard, meaning there do not currently exist algorithms that can solve this problem in polynomial time, but it can be solved reasonably fast in pseudo-polynomial time through dynamic programming for small input size. (**Pisinger**, **2004**) NP-hard problems such as travelling salesman problem and the Hamiltonian completion problem are more difficult to find improvements. (**Jooken**, **2021**)

The KP01 problem was first introduced by Dantzig (who is the founder of operational research) in 1957. In this research paper, Dantzig proposed a greedy approximation algorithm as an approach for the unbounded knapsack problem. (**Ezugwu**, **2019**)

2 Method

Construction graph shown in Fig. 1 is fully connected, to emphasize the decision that money bags can be accessed in any order.

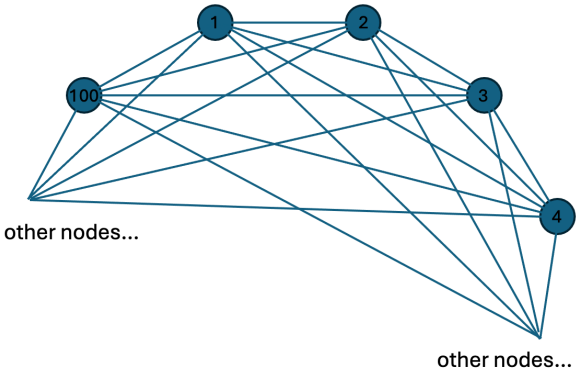

Figure 1: Fully Connected Construction Graph with 100 Nodes

Our pre-computed local heuristic values for each node j is defined as

$$\eta_j = \frac{v_j}{w_j}. \tag{3}$$

Our pheromone is initialised to 1 for all paths to allow for unbiased exploration in the beginning. Pheromone of path between node i and node j τ_{ij} is updated after each iteration by

$$\tau_{ij}(t+1) \leftarrow (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t), \tag{4}$$

where ρ is the rate of pheromone decay (evaporation rate), $0 < \rho < 1$, and

$$\Delta\tau_{ij}(t) = \sum_{i=1}^{p}(m_i) \tag{5}$$

is the sum of total pheromone deposited m by p ants in an iteration.

I'm going to try 3 different pheromone deposit parameters (m) and their performance will be evaluated in Section 3. The first parameter is $m = 1/w$, the intuition is that pheromone deposit m decrease as weight increases, so force ants favouring lighter weights. The second parameter I'm going to try is $m = \ln(v)/w$, this parameter take into consideration the value of the bag as well

as its weights. Since value ranges from 10 to 100, take the natural logarithm make sure pheromone doesn't overblown due to the value. The third parameter is $m = \ln(v)$.

Probability of ant moving from the current node i to the next node j is

$$p_{ij}(t) = \frac{[\tau_{ij}(t)]^{\alpha}(\eta_j)^{\beta}}{\Sigma_{h \in J_i^k}[\tau_{ih}(t)]^{\alpha}(\eta_h)^{\beta}}, \qquad (6)$$

where J_i^k is the set of available nodes, k in total.

To ensure optimal probability distribution for our algorithm, hyperparameters α, β, ρ will be tuned using iterative grid search method. Then optimal pheromone deposit parameter (m) will be assessed by conducting Kruskal-Wallis test to test for statistically significant difference in performance (fitness level). Finally, we will use grid search to find optimal population size p (also known as number of ants running concurrently within an iteration). Results for above procedure will be discussed in Section 3.

In the algorithm, we implement a tabu list for each ant to keep track of all the nodes it has visited. At each step "available nodes (J)", "cumulative weights" and "cumulative values" are updated to keep track of weight constraint and update the fitness value. The fitness of a solution path is its cumulative value, the best fitness is the highest cumulative value, as discussed in Section 1.

In the algorithm, I used `BoundedSemaphore` method from `threading` library, so that ants run in parallel in an iteration.

3 Results

We begin tuning hyperparameters by setting initial values to $\alpha, \beta = [0.5, 1.0, 1.5, 2.0]$ and $\rho = [0.1, 0.3, 0.5, 0.7, 0.9]$. There is no significant improvement from run 2 to run 5 (Table 1), which suggest the search space around this region could be rather flat. We select $\alpha = 1.3, \beta = 5.0, \rho = 0.90$ as our optimal hyperparameters.

To select the optimal parameter for pheromone deposit m defined in equation 5, the algorithm was run 200 times for each parameter. The first 10 results are displayed in Table 2.

By visual inspection, it's not difficult to tell the outputs did not conform to a normal distribution (see Fig. 2 in Section 4). Therefore, we conduct a Kruskal-Wallis test to determine whether there exist a statistically significant difference between the 3 groups of non-parametrically distributed outputs.

iteration (n) = 100				
	α	β	ρ	Best Fitness
Run 1	1.5	2.0	0.90	4517
Run 2	1.3	5.0	0.95	4526
Run 3	1.3	5.0	0.90	4524
Run 4	1.3	5.0	0.89	4524
Run 5	1.3	5.0	0.90	4519

Table 1: Hyperparameter Tuning using Iterative Grid Search Approach

$n = 100, \alpha = 1.3, \beta = 5.0, \rho = 0.9$					
	Run 1 to 5				
	1	2	3	4	5
$1/w$	4519	4518	4520	4520	4517
$\ln(v)/w$	4518	4517	4521	4519	4522
$\ln(v)$	4522	4521	4527	4515	4522
	Run 6 to 10				
	6	7	8	9	10
$1/w$	4522	4514	4513	4508	4491
$\ln(v)/w$	4521	4524	4524	4523	4522
$\ln(v)$	4527	4523	4527	4528	4527

Table 2: Fitness Levels under Different Pheromone Parameters (m)

The Kruskal-Wallis test statistic H is defined as follows:

$$H = \frac{N-1}{N} \sum_{i=1}^{k} \frac{n_i(\bar{R}_i - E_R)^2}{\sigma_R^2}, \qquad (7)$$

where N denotes the total sample size across all groups ($N = 600$), k denotes the number of groups ($k = 3$), $\bar{R}_i$ is the mean rank within each group, E_R is the expected rank given by $\frac{N+1}{2}$ and σ_R^2 is the rank variance given by $\frac{N^2-1}{N}$.

The test statistics is 1.70 and p-value is 0.43. Since p-value is greater than 0.05, we fail to reject the null hypothesis and conclude there is no statistically significant difference in output performance across the parameters. The parameter $m = \ln(v)$ is selected arbitrarily.

Now we perform grid search to find the optimal ant population size (p), we strike to find a balance between run time and minimal p required. From Table 3, we can see performance plateau after $p = 10$, as there is minimal imporvement in fitness level.

$n = 50, m = \ln(v), \alpha = 1.3, \beta = 5.0, \rho = 0.9$					
	1	10	50	100	200
Run 1	4501	4527	4528	4528	4528
Run 2	4519	4528	4528	4528	4528

Table 3: Fitness Levels under Different Ant Population Size (p)

We can conclude that the optimal parameters are $p = 10, m = \ln(v), \alpha = 1.3, \beta = 5.0, \rho = 0.9$. After running simulation for $10,000$ iterations the best fitness using these parameters is 4528.

4 Discussion and Further Work

Due to the stochastic nature of our meta-heuristic algorithm, we want to investigate the distribution of the outputs. The output (fitness level) was sampled 1,000 times and its frequency plot is shown in Fig. 2. The resulting distribution resembles a Beta distribution. Kolmogorov-Smirnov test was conducted to compare our sample distribution with 2,000 samples drawn from $Beta(5.9, 2.5)$. The visual reference to Kolmogorov-Smirnov test is shown in Fig. 3. We obtain test statistic 0.05 and p-value 0.08, indicating no statistically significant difference between the two distributions. Therefore, we conclude that our empirical distribution is reasonably similar to a Beta distribution.

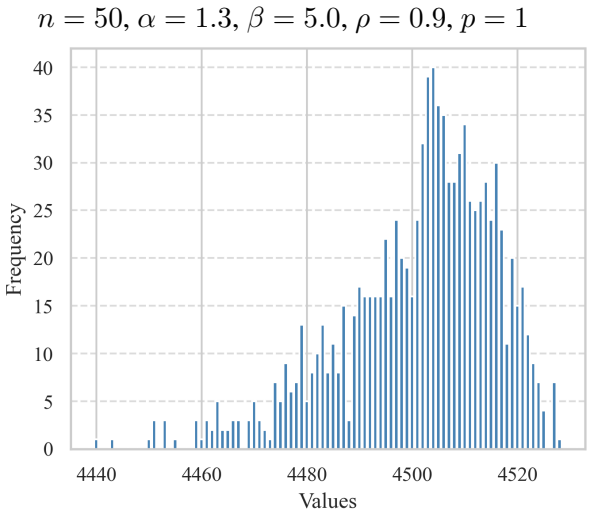

Figure 2: Frequency Distribution of Fitness Levels for 1000 Samples

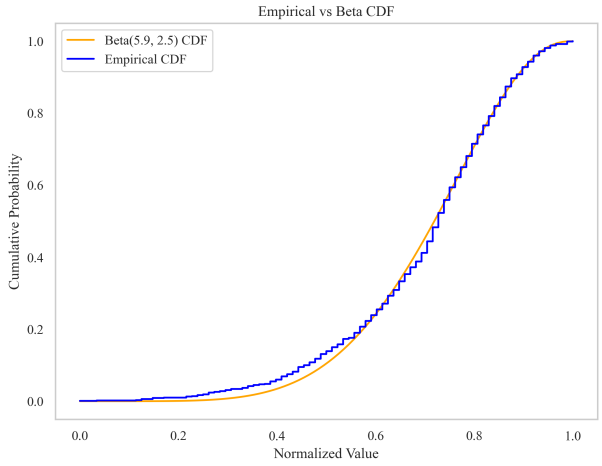

Figure 3: Cumulative Distribution Function Curve for Empirical Samples and Beta(5.9, 2.5) Samples

Question 1: Which combination of parameters produces the best results?

The optimal parameters are $p = 10, m = \ln(v), \alpha = 1.3, \beta = 5.0, \rho = 0.9$, assessed using grid search method and Kruskal-Wallis test, discussed in detail in Section 3.

Question 2: What do you think is the reason for your findings in Question 1?

The sample size for this KP01 problem is small, with only 100 nodes. As a result, the search space is small and the algorithm can find good solution in relatively short time. This explains why the performance plateaus after ant population size $p = 10$, as increasing the number of ants does not lead to significant improvements (Table 3).

The pheromone deposit m is set to $m = \ln(v)$, as discussed in detail in Section 3. Given that the value of v ranges from 10 to 100 in our problem, taking the natural logarithm ensures the pheromone doesn't overblown due to large value of v.

A larger value of ρ (rate of pheromone decay, also known as evaporation rate) prevents premature convergence to a local optimum due to the strong memory of past search history.

Question 3: How do each of the parameter settings influence the performance of the algorithm?

Suboptimal parameter settings leads to premature convergence or excessive running time, both of which are undesirable. For example, a lower evaporation rate $\rho = 0.5$ may lead the algorithm to converge prematurely, while a large ant popula-

tion size $p = 200$ can significantly delay the running time.

Question 4: Do you think that one of the algorithms in your literature review might have provided better results? Explain your answer.

Dynamic Programming is a common approach for solving the KP01 problem. Other proposed algorithms include Branch and Bound (BB), which is guaranteed to find an optimal solution given small problem size. Another approach is Simulated Annealing (SA), which is an effective heuristic for addressing highly non-linear models and noisy data, with the advantage of being able to escape local optima. (**Ezugwu**, **2019**) In conclusion, each algorithm have its unique advantage and the best performing algorithm is problem-specific.

Other ACO variant algorithms include Max-Min Ant System (MMAS), where only the best solutions in an iteration is allowed to add pheromone during the pheromone trail update. This can be written as

$$\tau_{ij}(t+1) \leftarrow (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}^{best}(t),$$

where $\tau_{ij}^{best}(t) \propto \frac{1}{f(s^{best})}$, f is the solution cost. A good approach is to adopt a mixed strategy that we choose iteration-best by default s^{ib} and choose global-best only every fixed number of iterations. (**Stutzle**, **2000**)

Another ACO variant is the Elitist Ant System (EAS). In EAS, pheromone trails are updated after each iteration based on all ant paths, with an additional amount by the global best path. (**Brownlee**, **2024**)

References

Pisinger, D. (2004). Where are the hard knapsack problems? *Computers & Operations Research*, 32 (2005), pp.2271–2284. doi:https://doi.org/10.1016/j.cor.2004.03.002.

Jooken, J., Leyman, P. and De Causmaecker, P. (2021). A new class of hard problem instances for the 0–1 knapsack problem. *European Journal of Operational Research.* doi:https://doi.org/10.1016/j.ejor.2021.12.009.

Ezugwu, A.E., Pillay, V., Hirasen, D., Sivanarain, K. and Govender, M. (2019). A Comparative Study of Meta-Heuristic Optimization Algorithms for 0 – 1 Knapsack Problem: Some Initial Results. IEEE Access, [online] 7, pp.43979–44001. doi:https://doi.org/10.1109/ACCESS.2019.2908489.

Dorigo, M. and Gambardella, L.M. (1997). Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation, [online] 1(1), pp.53–66. doi:https://doi.org/10.1109/4235.585892.

DATAtab (2021). Kruskal-Wallis-Test (Simply explained). [online] YouTube. Available at: https://www.youtube.com/watch?v=l86wEhUzkY4 [Accessed 10 Nov. 2024].

Wikipedia Contributors (2024). Kruskal-Wallis test. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/Kruskal%E2%80%93Wallis_test [Accessed 10 Nov. 2024].

Matthew E. Clapham (2016). 10: Kolmogorov-Smirnov test. [online] YouTube. Available at: https://www.youtube.com/watch?v=ZO2RmSkXK3c [Accessed 10 Nov. 2024].

Stutzle, T. and Hoos, H.H. (2000). – Ant System. Future Generation Computer Systems, 16(8), pp.889–914. doi:https://doi.org/10.1016/s0167-739x(00)00043-1.

Brownlee, J. (2024). Elitist Ant System. Available at: https://algorithmafternoon.com/ants/elitist_ant_system/ [Accessed 14 Nov. 2024].