The Team Optimisation Problem

Solved by Ant Colony Optimisation

A Detailed Study
by Siyamthanda Ndlovu

Table of Contents

Introduction

The Team Orienteering Problem (TOP) is a combinatorial optimisation problem derived from the Orienteering Problem (OP), extended to address scenarios involving multiple agents. This project aims to design, implement, and evaluate an Ant Colony Optimisation (ACO) algorithm that maximises the total collected reward from a set of locations, subject to route constraints including travel time and node visitation limits. The approach has applications in logistics, drone delivery, and search-and-rescue operations.

Problem Formulation

Description of Nodes, Vehicles, Depot, and Rewards

This iteration of TOP involves a set of *n* nodes, each representing a unique location with an associated score. A special node (node 0), the depot, serves as the starting and ending point for *m* vehicles. Each vehicle can visit a subset of nodes.

Constraints

- Each route must not exceed a maximum travel time or distance limit (*Tmax*).
- Each node may be visited by at most one vehicle.
- All vehicles must start and end their journeys at the depot.

Objective Function

The goal is to maximise the total score collected by all vehicles while satisfying the above constraints. Given a graph G = (V,A) where:

- V is the set of vertices (locations).
- A is the set of edges (paths between locations).
- tij is the travel time from vertex i to j
- si is the score at vertex i.

The mathematical formulation requires vehicles to start at vertex v1 and end at vertex vN, with total travel time not exceeding Tmax. The objective is to maximise the sum of scores collected by visiting selected vertices, represented as:

$$\sum s_i \times x_{ij}$$

where:

- s_i denotes the score associated with vertex i
- x_{ij} is a binary decision variable equal to 1 if vertex i is immediately followed by vertex j in the path, and 0 otherwise.

More on modelling this problem is in Vansteenwegen et al., *The Orienteering Problem: A Survey, EJOR*, 209(1), 2011

Ant Colony Optimisation (ACO) Design

Algorithm

The implementation follows the ACO framework proposed by Ke et al. (2008), "Ants can solve the team orienteering problem". The core algorithm operates as follows:

Initialize the parameters, heuristic information η and the pheromone trails τ
$s_{ib} \leftarrow Null$
$s_{gb} \leftarrow Null$
$N_{ni} \leftarrow 0$
$iteration \leftarrow 0$
while *iteration* is less than N_C do
  for $i = 1$ to n_a do (n_a is the number of ants)
    $s_i \leftarrow$ **ConstructSolution**(τ, η) (see section 3.2)
    $s_i \leftarrow$ **LocalSearch**(s_i) (see section 3.4)
  end for
  $s_{ib} \leftarrow \arg\max(F(s_1), F(s_2), \cdots, F(s_{n_a}))$
  if $F(s_{ib}) > F(s_{gb})$
    $s_{gb} \leftarrow s_{ib}$
    $N_{ni} \leftarrow 0$
  else
    $N_{ni} \leftarrow N_{ni} + 1$
  end if
  PheromoneUpdate($\tau, s_{ib}, s_{gb}, N_{ni}$) (see section 3.3)
  $iteration \leftarrow iteration + 1$
end while

While the termination condition is not met (maximum iterations):

- Each ant constructs a solution (routes for all vehicles)
- Solutions are evaluated based on total collected score
- Pheromone trails are updated based on solution quality
- The best solution is tracked and preserved

Implementation and Experimental Setup

Problem Instances

Predefined TOP files. The format of the text file provides the number of nodes, number of vehicles and maximum distance. Euclidean distance is subsequently utilised to generate the distance matrices for the fully connected graph. An example of a valid text file is presented below.

```
n 33
m 2
tmax 7.5
19.100     24.300     0
12.600     24.900     20
14.400     28.000     20
16.900     28.100     20
20.700     28.200     20
12.500     26.600     20
21.800     27.300     20
12.500     22.600     20
22.500     17.000     30
19.900     15.000     30
14.900     15.100     30
11.500     18.600     30
12.400     29.800     30
17.800     28.100     30
9.100      29.800     40
10.000     32.600     40
13.900     33.100     40
19.950     10.300     40
15.200     8.000       40
14.700     31.200     50
7.400      36.500     50
21.000     25.500     50
18.000     25.300     10
19.500     24.700     10
21.400     21.800     10
16.000     21.400     10
18.650     26.200     10
17.900     28.900     10
14.300     19.900     20
17.000     19.000     20
10.800     21.000     20
15.700     23.700     10
18.200     24.000     0
```

Local Search

Iterated local search was initially implemented and tested but subsequently omitted from the final algorithm due to its failure to yield significant improvements—indeed, results were often markedly inferior to those achieved without it.

Faster metaheuristics such as simulated annealing were considered but ultimately deprioritised in favour of focusing development efforts on optimising the core ACO implementation.

Algorithm Implementation

Each vehicle's route is represented as an ordered list of nodes, beginning and ending at the depot (node 0). This representation was adopted to ensure route feasibility (guaranteeing return to depot), simplify distance calculations through explicit inclusion of return trips, and maintain alignment with real-world logistics scenarios where vehicles typically return to a central hub.

Pheromone Initialisation

Initial pheromone values were uniformly set to 0.1 across all edges. This relatively low value was selected to encourage early-stage exploration of the solution space whilst avoiding premature convergence—a practice well-established in ACO literature where uniform, modest pheromone levels are standard for initialisation.

Pheromone Update Strategy

The pheromone update mechanism operates through two complementary processes:

Local Update (per iteration): Better solutions deposit proportionally more pheromone, with the inverse of cost ensuring shorter, higher-reward routes receive stronger reinforcement. This approach directly incentivises the discovery of efficient, high-value routes.

Evaporation: Global evaporation of pheromone values occurs after each iteration, with values immediately updated upon discovery of new best solutions. This mechanism prevents stagnation and maintains healthy exploration-exploitation balance throughout the search process.

Node Selection

The node selection strategy employs a benefit-to-cost heuristic expressed as `ηij = sj / cij`, where `sj` represents the reward at node `j` and `cij` the travel cost from node `i` to `j`.

This ratio-based approach was selected for its intuitive balancing of reward maximisation against travel cost minimisation, promoting efficient route construction through prioritisation of high-value, easily accessible nodes.

Route Construction

The route construction process incorporates a critical feasibility check before adding any node: `current distance + edge cost + return to depot cost ≤ max distance`. This constraint was implemented to ensure vehicles can always return to the depot, prevent consideration of infeasible partial solutions, and maintain consistency with real-world operational limitations.

Parameter Settings

Parameter tuning was conducted through empirical testing to optimise the balance between exploitation and exploration. The experimental design included systematic variation of:

- Number of iterations: 10, 20, 30
- Number of ants: 5, 10, 15, 20
- Pheromone importance weight (α): 1, 2
- Heuristic information weight (β): 1, 2
- Initial pheromone value (τ_0): 0.1
- Pheromone reinforcement strategy: With and without reinforcement of best solutions

Seed Configuration

To ensure robust statistical evaluation, each parameter combination was tested with 10 different random seeds, with 10 independent runs per seed-parameter combination - yielding 100 total executions per parameter set. This comprehensive approach was adopted to account for the inherently stochastic nature of the ACO algorithm.

Execution Environment

All experiments were executed on a single machine, with the ACO implementation written in Java. The experimental harness automatically tracked and recorded solution quality, execution time, and convergence behaviour for each run. Results were stored in a structured CSV format for subsequent analysis and visualisation.

Data Collection

For each experiment, the following data was collected:

1. Solution quality (total route cost)
2. Iteration where the best solution was found
3. Execution time in microseconds
4. The actual solution (routes assigned to vehicles)

Result Analysis

For each experiment, the following data was collected:

- Solution quality (total route cost)
- Iteration where the best solution was found
- Execution time in microseconds
- The actual solution (routes assigned to vehicles)

Results

Top 2 results are shown. Analysis was conducted on the top 5% of results. The averages of those results were selected and are highlighted in bold in the averages row in the table that follows.

#	Instance	Solution	Seed	Score	Cost	Duration
1	p3.2.a	0-23-21-26-22-0 | 0-32-31-0	90	90	10	920
		0-32-31-0 | 0-23-21-26-22-0	50	90	10	904
		Average:	**83.96**	**90**	**10.75**	**1339.56**
2	p3.2.c	0-22-13-3-27-0 | 0-23-21-6-4-26-0	100	180	14	1006
		0-21-6-4-26-23-0 | 0-22-3-13-27-0	100	180	15	1128
		Average:	**89.79**	**180**	**15.79**	**1721.46**
3	p3.3.a	0-23-32-0 | 0-21-0 | 0-22-26-0	40	80	8	853
		0-32-22-0 | 0-23-21-0 | 0-26-0	50	80	7	744
		Average:	**71.67**	**80**	**7.1**	**608.5**
4	p3.3.c	0-23-21-6-0 | 0-22-31-0 | 0-13-26-0	80	140	19	626
		0-23-21-6-0 | 0-22-31-0 | 0-26-13-0	90	140	19	863
		Average:	**82.5**	**140**	**18.75**	**2115.71**
5	p3.4.s	0-23-21-6-4-13-3-19-16-0 | 0-25-28-11-30-7-1-5-0 | 0-31-29-9-8-24-0 | 0-26-22-2-12-14-0	100	590	85	2114
		0-23-21-6-4-19-16-32-0 | 0-26-13-3-2-12-14-5-0 | 0-22-31-1-7-30-11-28-0 | 0-24-8-9-29-25-0	90	590	82	3350
		Average:	**95.21**	**570.21**	**83.81**	**8561.67**
6	p3.4.t	0-23-21-6-4-13-3-19-16-2-0 | 0-24-8-9-10-29-0 | 0-22-26-5-1-7-30-11-28-25-0 | 0-14-15-12-0	100	650	92	2046

		0-23-21-6-4-13-3-19-16-2-0 | 0-26-22-5-1-7-30-11-28-25-0 | 0-14-15-12-0 | 0-29-10-9-8-24-0	100	650	91	4832
		Average:	**96.66**	**619.17**	**90.96**	**9121.27**
7	p7.2.a	0-101-39-0 | 0-29-0	70	30	18	2490
		0-101-39-0 | 0-29-0	70	30	18	2093
		Average:	**56.04**	**30**	**18**	**2340.08**
8	p7.2.m	0-101-29-30-20-9-93-15-52-10-33-17-47-45-3-0 | 0-39-41-21-14-57-100-8-56-11-99-74-60-42-1-12-36-22-13-59-0	80	636	236	43767
		0-42-23-12-36-22-60-11-56-8-14-21-13-59-57-58-100-1-0 | 0-9-93-20-40-16-48-49-27-91-90-26-28-30-29-39-41-0	100	635	241	43383
		Average:	**91.46**	**575.81**	**236.46**	**30940.04**
9	p7.3.h	0-101-29-30-9-93-20-0 | 0-39-41-21-14-57-59-13-0 | 0-100-99-11-56-8-98-0	80	345	146	16025
		0-8-56-11-99-74-101-0 | 0-29-30-20-9-93-0 | 0-39-41-21-14-57-59-13-0	90	342	144	19119
		Average:	**81.88**	**321.39**	**144.21**	**32566.31**
10	p7.4.g	0-30-20-28-0 | 0-101-39-41-21-14-0 | 0-29-81-55-0 | 0-3-0	90	201	128	25939
		0-101-29-39-41-0 | 0-30-20-28-0 | 0-21-14-57-0 | 0-3-0	90	197	127	4525
		Average:	**81.25**	**186.52**	**124.27**	**30444.5**
11	p7.4.q	0-101-39-41-100-8-56-11-60-99-74-98-59-13-21-14-57-0 | 0-29-30-20-9-93-51-15-52-10-33-38-0 | 0-55-24-1-94-23-42-0 | 0-53-6-5-25-43-18-0	90	693	301	29690

		0-101-29-3-18-43-6-53-24-55-0 | 0-30-9-51-15-52-10-33-17-80-38-0 | 0-39-41-21-14-57-59-100-8-56-11-60-99-74-13-98-0 | 0-28-20-50-93-48-0	90	692	309	49580
		Average:	**89.58**	**634.62**	**306.13**	**43996.16**

Anomalies in Results

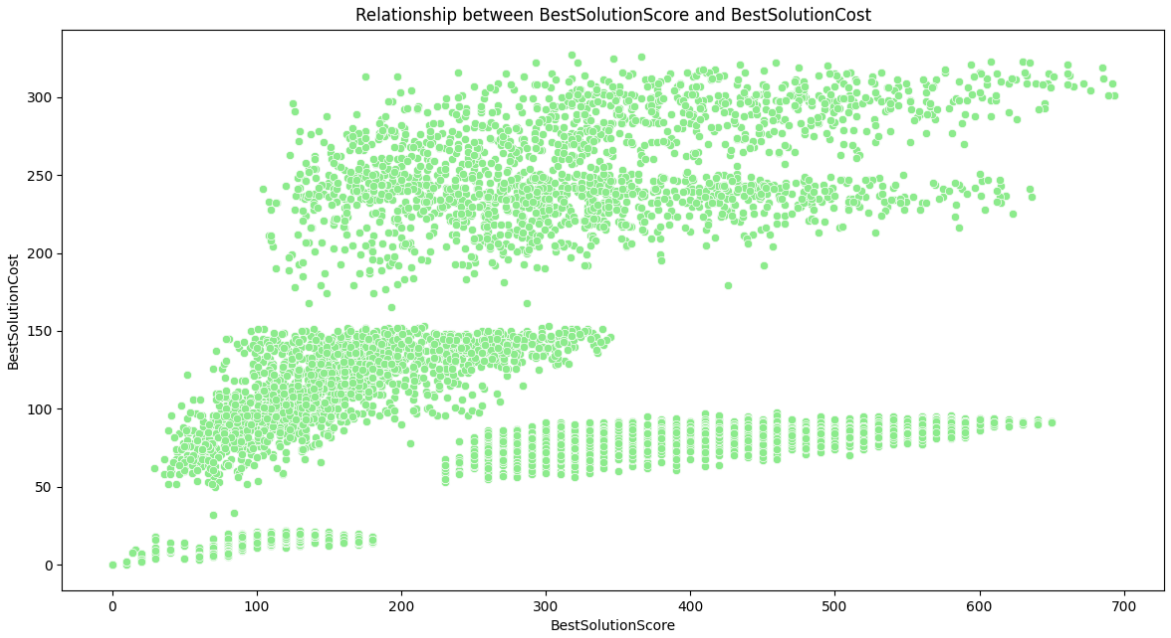
One specific data point for the 102-node problem (NumberOfAnts=10, Seed=20, EnforceStrongSolution=enforced) shows a BestSolutionCost of 27.0, which is significantly lower than typical costs for similar configurations (average cost around 82 for this group). The corresponding solution string `0-0 | 0-101-0 | 0-21-14-0 | 0-0` indicates only 3 nodes visited across 4 vehicles (two empty routes), suggesting a potentially incomplete or erroneous solution run, or a data recording error, as a cost this low for visiting 3 nodes in a 102-node problem with MaxDistance 35.0 seems highly improbable.

Analysis

The analysis focused on the top 5% of results, which were selected based on their performance. Only these high-performing results were used to carry out a detailed evaluation.

Costs of Best Solutions

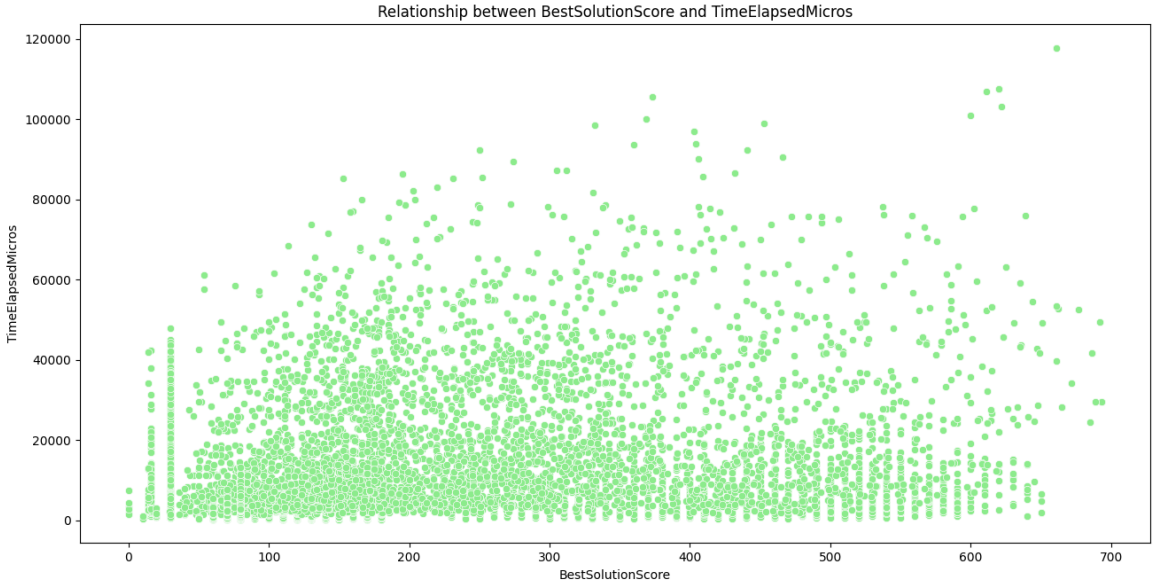
Looking at the costs versus the rewards of the best performing results, no clear pattern was found, but four distinct cluster values were identified when plotting the relationship between the scores of the best solutions versus the cost of those solutions. The top cluster spans through a large part of the x-axis and the top part of the y-axis.



Scatter plot 1: The rewards from the best solutions plotted against the costs of those solutions

Time to Best Solutions

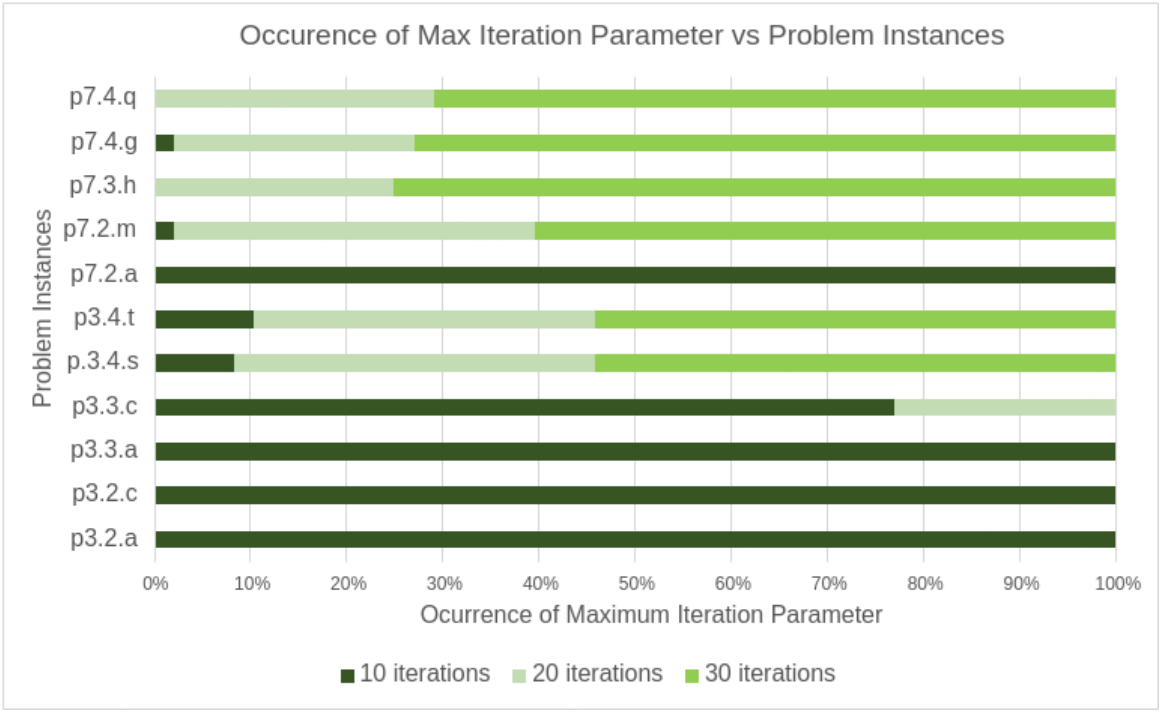
The time it took to get to the best solutions was tracked in microseconds. Across the 11 problem instances, the vast majority of the solutions fall under the cost of 60,000 microseconds, and the values cluster towards the bottom. All the values become considerably less dense as the particles move upwards towards the top of the graph (i.e., as time increases, there are fewer values plotted).



Scatter plot 2: Rewards score of the solutions compared to the time taken to get the solution

Maximum Iteration Analysis

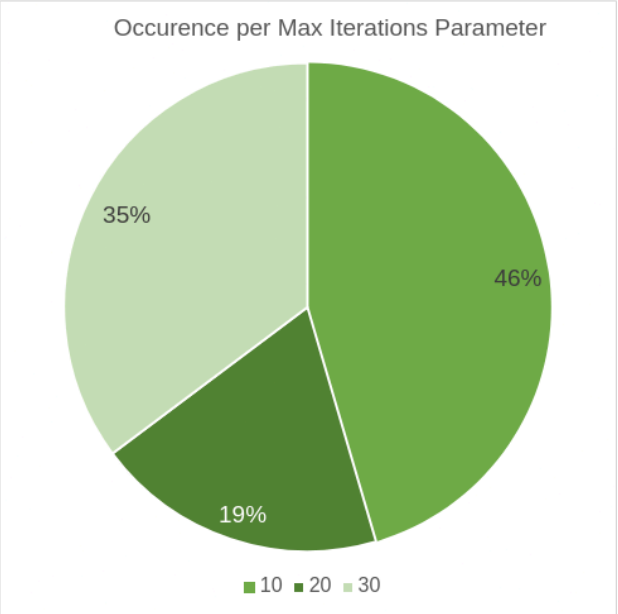
During the experiment, the maximum iteration values were set as 10, 20 and 30. Looking at the top results, the number of times that each of these values appeared in the top results was analysed. This is summarised in the following chart:



Bar chart 1: Each instance is listed along the y-axis and the percentage of the best results for that instance is shown

In 4 out of 11 instances, only maximum iterations of 10 were present, making it the most common iteration value. The seed that occurred most frequently was at 30 iterations.

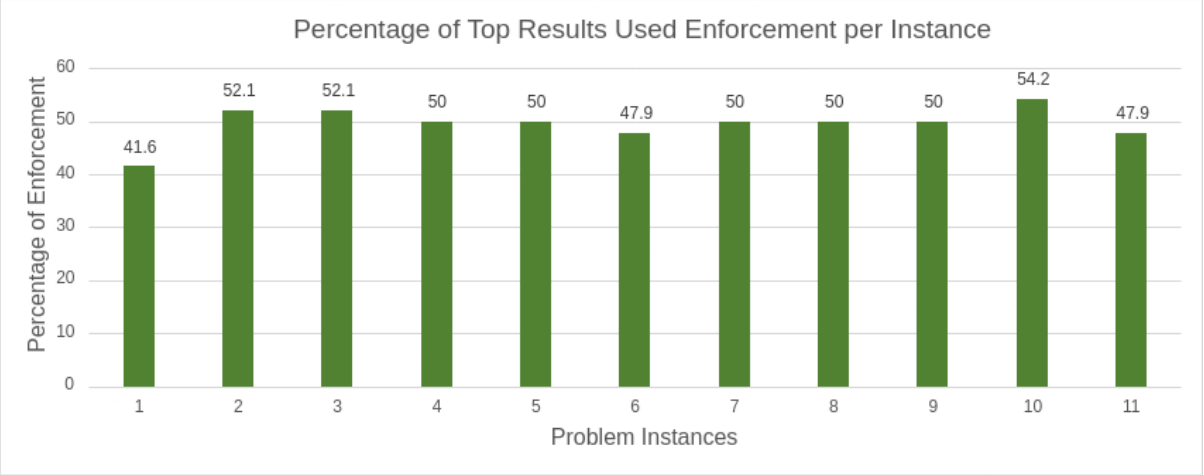
Based on this data, there appears to be no specific correlation between the number of iterations and the quality of solutions. No clear patterns were apparent. The occurrence frequency of iterations 10, 20, and 30 is summarised in the chart on the right.



Pie chart 1: Parameter values which occurred the most

Enforcement Parameter Analysis

This parameter was applied when an additional pheromone value of 0.5 would be added to the trails that showed improvements, in an attempt to emphasise trails that were successful. The differences are better explained in the following graph which shows the slight variations between the enforced and non-enforced solutions. The chart shows what percentage of the top solutions are enforced per problem instance:



Bar chart 2: The percentage of the top results in each instance that used the enforcement of best solutions

Whether or not the best trails were enforced seems to have an almost identical distribution. Enforced solutions performed only slightly better than unenforced solutions, but the difference was minimal. The following chart shows how closely the two parameters were in terms of performance:

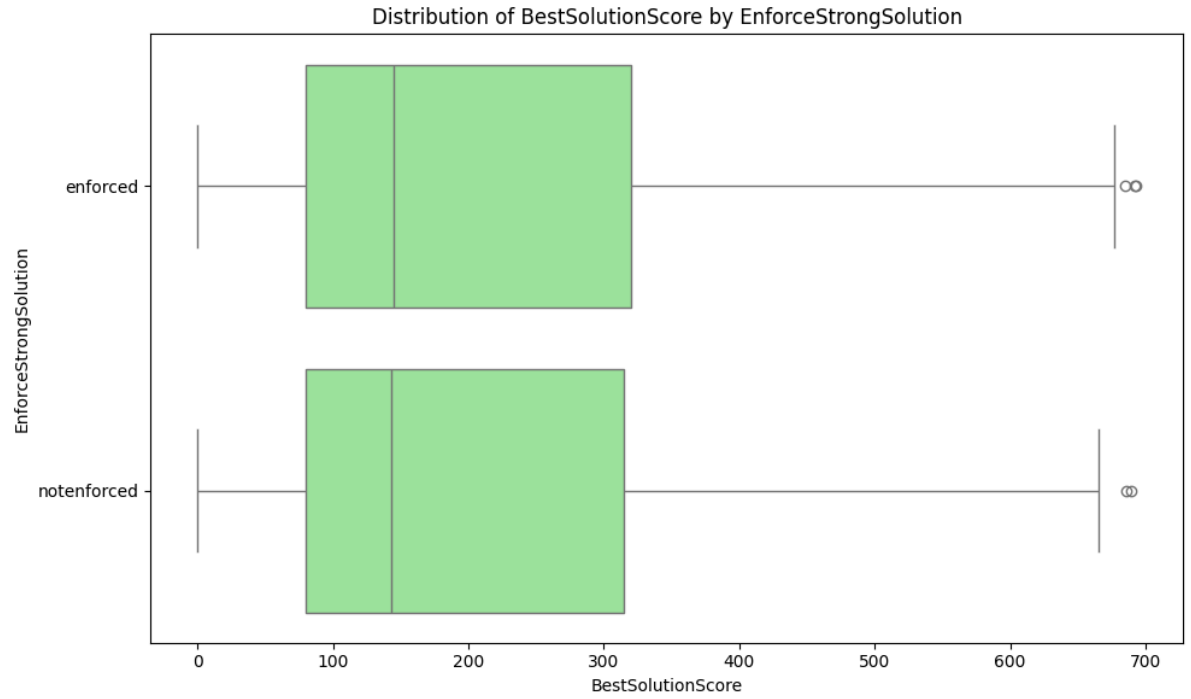


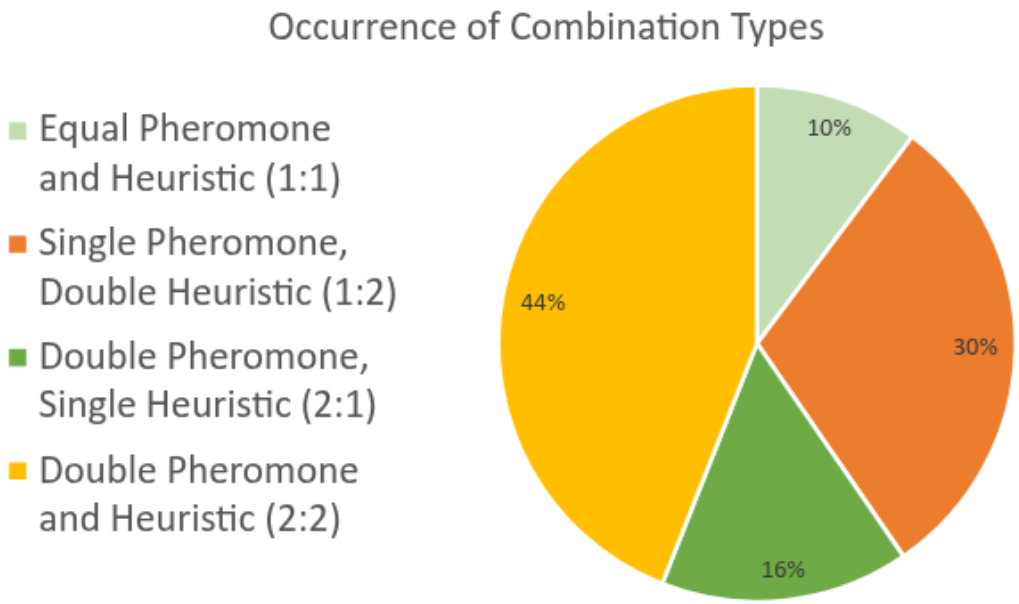
Chart 1: Distribution of reward scores grouped by enforced and unenforced

Pheromone and Heuristic Weight Combination Analysis

As elaborated in the setup sections of this report, in the decision of what nodes to pick next, the pheromone and heuristic values are taken into consideration. For the experiment, the weights of the pheromone and the heuristic were varied.

A ratio of 1:1 meant the heuristics and pheromone had the same weights, whilst 1:2 meant weight 1 for pheromone and 2 for heuristic, and so on.

The emphasis on both (i.e., giving both the pheromone and the heuristic a weight of 2) had the highest occurrence among the top results, indicating that greater emphasis on both factors led to better results.

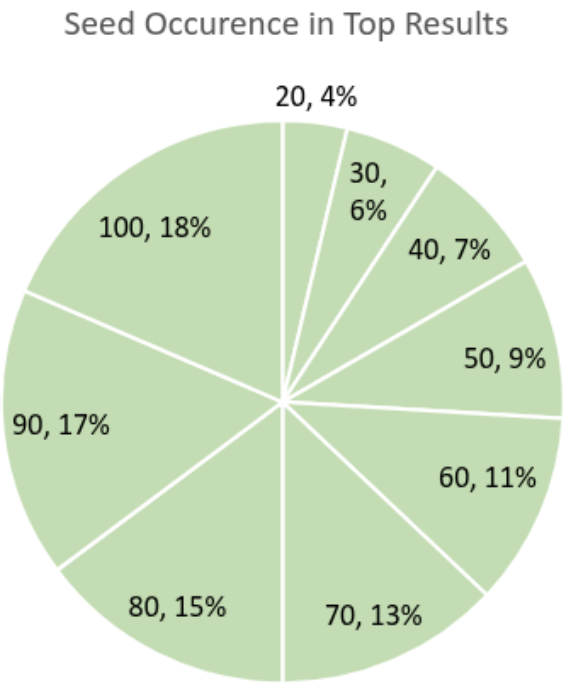


Pie chart 2: Occurrence of weight ratios within top performing results

Seed Analysis

The seed in the implementation of the algorithm was used to control the roulette wheel selection. The seed was an integer between 10 and 100. The higher the seed, the more randomised the roulette selection.

Interestingly, seed 10 did not appear at all in the best results that were analysed. The top performing seeds were the higher ones, particularly 100, with 90 showing almost identical performance.



Pie chart 3: Occurrence of seeds among best results

Furthermore, there is a clear correlation between higher seed values and how frequently that seed occurs in the best results, which suggests a correlation between seed value and higher performance.

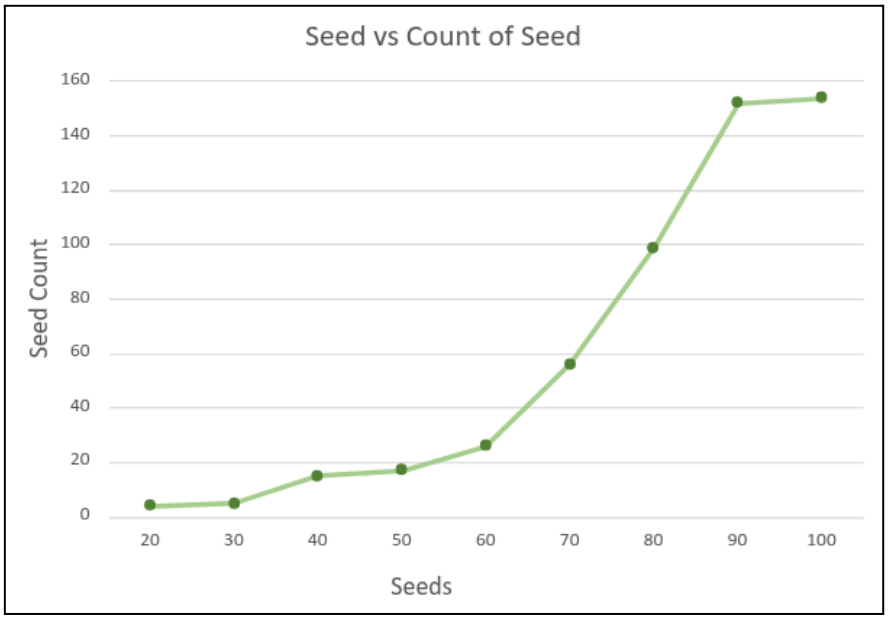
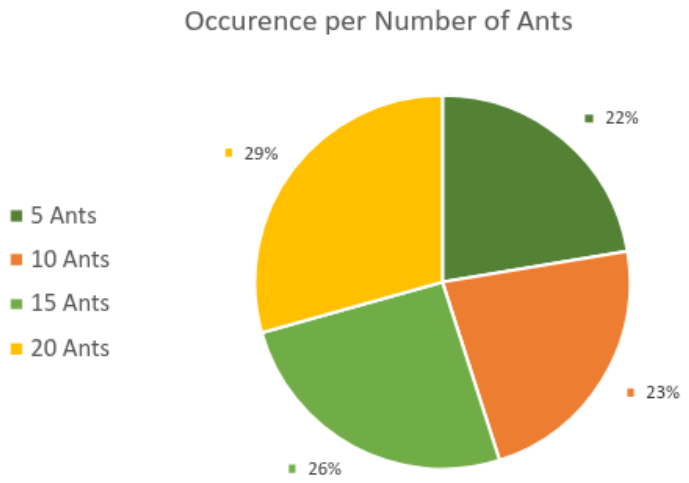


Chart 2: Occurrences of seeds versus the seed value

Analysis of Ants Parameter

The four values used for the number of ants parameter were 5, 10, 15 and 20. Looking at how often each of these values appeared in the analysed results (the best-performing ones), the performance of these values seems to be similar, ranging from 22% to 29% occurrence rates.



Pie chart 3: This shows how often each of the parameter values occurred in the analysed results

Looking at the data, however, there is yet another clear correlation between the occurrence of the value and the value of the parameter itself. This means that the higher the number of ants, the more successful the solutions generated.

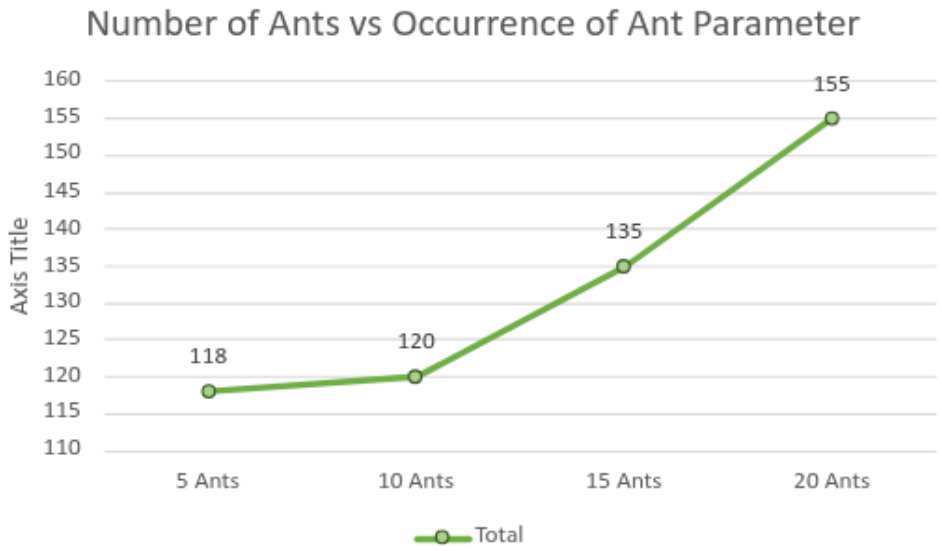


Chart 3: Number of ants versus the occurrence of the ant parameter

Analysis of 102-Node and 33-Node Problems

Key Insights from 102-Node Problem

For the larger problem instance (102 nodes, 4 vehicles):

- Increasing the number of ants generally leads to a higher best solution score (implying more nodes visited or better solution quality) but also significantly increases the time elapsed in microseconds.
- The best solution cost (total distance) also tends to increase with more ants, suggesting longer routes are explored to visit more nodes.
- Enforcing stronger solutions seems to have a less pronounced effect on average trends compared to the number of ants, although there might be subtle differences in distribution or specific cases.

Key Insights from 33-Node Problem

For the smaller problem instance (33 nodes, 2 vehicles):

- Increasing the number of ants from 5 to 10 shows a slight increase in best solution score and time elapsed in microseconds.
- The effect on best solution cost is less pronounced but shows a slight average increase.
- Enforcing stronger solutions appears to influence performance. Runs without enforcement tend to have slightly higher best solution scores and lower time elapsed compared to enforced runs with the same number of ants.

Observed Trends

For 102 nodes (4 vehicles, MaxDistance 35.0):

- Increasing ants correlates with higher average score
- Increasing ants correlates with higher average cost
- Increasing ants correlates with significantly higher average computation time

For 33 nodes (2 vehicles, MaxDistance 7.5):

- Increasing ants correlates with slightly higher average score
- Increasing ants correlates with slightly higher average cost
- Increasing ants correlates with higher average time
- Configurations without enforcement tend to yield slightly higher scores and lower computation times compared to those with enforcement.

Future Improvements

Implementation Improvements

Local Search Implementation

The addition of a local search procedure could significantly improve the quality of solutions. Integrating a local search phase before pheromone updates would allow for the generation of more refined candidate solutions. Combining the global search capabilities of the ant colony optimisation framework with the intensification power of local search techniques would help to exploit promising regions of the solution space more thoroughly.

Experimental Improvements

Variation of Initial Pheromone Deposits

Varying the amount of initial pheromone deposits could provide insights into its influence on solution quality. Although enforcing pheromone reinforcement on the best solutions did not yield significant improvements, altering the starting pheromone values might affect the early search dynamics and, consequently, the overall performance.

Increase Number of Ants Range

Using more than 4 ant parameter values (5, 10, 15, 20) could be implemented to confirm that there is indeed an improvement with increased ant numbers. In the results, the top solutions showed that the higher number of ants, such as 15 and 20, occurred more frequently than the lower number of ants.

The correlation was slight but present - increasing the number of ants to be between 5 and 20 (e.g., 5, 6, 7...) could provide more confidence in the result. Examining whether the results would continue to improve after using 20+ ants could also provide valuable insights into the scalability of the algorithm and potential performance plateaus.

Tracking convergence

In the experiments, the iteration in which the best solution was first discovered is tracked. However, this is not sufficient to analyse convergence rates comprehensively. A more robust approach would be to track each new solution discovered in every iteration and observe how the rewards of these solutions change over time.

Analysing how much each score deviates from previous iterations would provide a great way of measuring how quickly the algorithm converges to optimal solutions. This data could also help identify premature convergence issues and inform adjustments to the algorithm parameters.

Varying Weights

The weights parameter produced straightforward results - increased emphasis appeared to yield better performance. To establish greater confidence in these findings, more combinations of weights should be tested. This would enable a more nuanced understanding of the relationship between pheromone and heuristic influence.

Additionally, examining non-linear weight combinations or adaptive weight strategies throughout the search process could potentially reveal more sophisticated patterns and lead to improved algorithm performance. The current findings suggest that higher weights produce better results, but determining if there is an optimal ceiling beyond which further increases yield diminishing returns would be valuable.

Node Selection Methods

Experimenting with different node selection methodologies would be beneficial in determining if alternative approaches could yield superior results. The current implementation relies on a roulette wheel selection mechanism, but other methods such as rank-based selection, tournament selection, or epsilon-greedy approaches could be evaluated.

Each selection method introduces different exploration-exploitation balances that might be more suitable for particular problem instances. A comprehensive comparison of these methods would provide insight into which selection strategy is most effective for the Team Orienteering Problem and under what conditions.

Conclusion

The analysis of the ant colony optimisation (ACO) solution data reveals distinct performance characteristics for two different vehicle routing problem (VRP) sizes. For the larger problem, increasing the number of ants improves solution quality (score) but at the cost of increased computation time and potentially higher total distance.

For the smaller problem, similar trends are observed, with solutions without enforcement appearing slightly more efficient. The results show significant variability influenced by the random seed. A potential anomaly was identified in the cost data for one specific run in the larger problem set. Predictive analysis is limited to describing these observed trends due to the fixed nature of most problem parameters in the dataset.

The ACO algorithm effectively addressed the constraints of the team orienteering problem and produced competitive solutions. Its ability to balance reward maximisation with travel feasibility shows promise, though further refinement, particularly in parameter tuning, could yield even better results in future work.

References

(1) Pieter Vansteenwegen, Wouter Souffriau, Dirk Van Oudheusden, The orienteering problem: A survey, European Journal of Operational Research, Volume 209, Issue 1, 2011
(https://www.sciencedirect.com/science/article/pii/S0377221710002973)

(2) Liangjun Ke, Claudia Archetti, Zuren Feng, Ants can solve the team orienteering problem, Computers & Industrial Engineering, Volume 54, Issue 3, 2008
(https://www.sciencedirect.com/science/article/pii/S0360835207002161)