Optimizing Telescope Slewing through the Traveling Salesman Problem using the Ant Colony Optimization algorithm

Proag Tanvee, 2422814
University of Mauritius

Abstract

In the navigation of celestial bodies, telescope slewing is similar to the Traveling Salesman Problem (TSP) where the objective is to minimize the distance traveled by finding the most efficient sequence of observations to reduce the telescope's positional adjustments. Traditional algorithms for the TSP (like the Greedy Search) serve as benchmarks for evaluating the effectiveness of the variants of Ant Colony Optimization (ACO). The results of Ant System (AS), ASElite (Ant System Elite) and MMAS (Max Min Ant System) are compared and evaluated. Hyperparameter tuning is done using Bayesian Optimisation and the results of the tuned and untuned parameters are compared. Some strategies yield a significantly shorter path in search spaces that are larger. However, this improvement comes at the expense of increased computational time.

1. Problem Statement

1.1 Telescope Slewing

With advanced telescopes conducting large sky surveys, time efficiency is important to maximize observation periods and minimize downtime caused by mechanical movements. The goal of slewing optimization is to minimize the time taken for a telescope to move between targets. By treating each celestial body as a node in a TSP, an optimal path can be derived, similar to the optimization of travel routes between cities. Astronomers benefit from this decreased total time spent moving between stars as it allows for more observations within a given time frame.

1.2 Time Complexity

The Traveling Salesman Problem (TSP) is a combinatorial optimization problem focused on finding the shortest route through a set of cities. As an NP-hard problem, it is computationally challenging especially with an increasing number of cities. No polynomial-time algorithm exists to solve TSP in all cases. TSP can be approached in non-deterministic polynomial time or heuristics and approximation algorithms (like the Ant Colony Optimisation) can be used to achieve near-optimal results efficiently.

2. Literature Review

Related past works

1. *GalaxyTSP: A New Billion-Node Benchmark for TSP* by *MIT* and *Columbia University* Iddo Drori (2020) present a divide-and-conquer approach that runs LKH on smaller tiles of the graph with 1,691,937,135 nodes to make space telescope target observation slew-optimized. ***Algorithm*** - Brute Force search for regions with fewer than 10 targets and Simulated Annealing for more dense regions, Lin-Kernigham Hergaus with divide-and-conquer

2. *Observations of Transiting Exoplanets with the James Webb Space Telescope (JWST)* by Beichman (2014), requires multi-objective optimization as it needs to balance its objectives between fuel efficiency, thermal stability and observation priority etc. By accommodating multiple objectives and handling complex constraints, GTSP enables JWST to maximize its scientific output while minimizing operational costs. ***Problem Formulation*** - GTSP (General Traveling Salesman Problem)

3. In *Astro-TSP: Traveling Salesman Problem Based Solutions for Scheduling Astronomical Observations* Humphries (2023) from the *California State Polytechnic University*, Right Ascension and Declination are treated like rectilinear coordinates x and y to calculate travel time between telescope pointing. Despite the non-deterministic nature of Genetic Algorithms, it regularly outperformed the Simple Sort and Look Ahead Greedy methods with the drawback of longer run times. ***Algorithm*** - Simple Sort, Look Ahead Greedy, Genetic Algorithm

4. In *Optimal Trajectory Control of an Occulter-Based Planet-Finding Telescope* by Egemen Kolemen (2007) at the *Princeton University*, the trajectory optimization of the occulter motion between imaging sessions of different stars is solved as a Time-Dependent TSP, discrete optimisation problem. Branch-and-Cut had a long computational time and the results were far from optimal, but the results got better when Tabu Search was combined with it. ***Algorithm*** - Branch-and-cut, Tabu Search

5. In the article *Algorithm of the Gods* published in *The Scientific Journal*, when Carlson (1997) had to use a 40-year-old telescope that sent its drive system into shock for large excursions, he concluded that SA is capable of finding near-optimal solutions in a reasonable amount of time. ***Algorithm*** - Simulated Annealing - Metropolis

6. *Particle Swarm Optimization: Method and Applications of Swarming in System Design* by Rania Hassan (2005) from *MIT*, examines the claim that PSO has similar effectiveness to GA but offers significantly better computational efficiency, validated through formal hypothesis testing and statistical analysis in telescope array configuration and spacecraft reliability-based design. PSO is observed to have poor repeatability in terms of finding optimal solution and computational cost. ***Algorithm*** - Genetic Algorithm, Particle Swarm Optimization

7. *An Improved Ant Colony Algorithm and Its Application In Routing Computation of Satellite Networks* Fei Long (2006), successfully avoid local optimization and generate a much shorter convergence time using the Ant system with elitist (ASelite) strategy in the routing optimization of satellite networks. ***Algorithm*** - Modified Ant Colony Optimization

3. Data

The Galaxy TSP dataset is based on the European Space Agency (ESA) Gaia space observatory, launched in December 2013. It is continuously scanning the sky with the goal of mapping the position and other properties of over a billion stars. Here the SsoObservation of Gaia data (published in 2023) is used: the Gaia data is downloaded from the online ESA Gaia Archive from which is extracted the on-sky position of the celestial bodies, consisting of two angular coordinates: Right Ascension (RA) and Declination (Dec). Those coordinates of a position on the sky are conceptually similar to the latitude and longitude coordinates of a location on the Earth. RA ranges between 0 to 360 degrees, and Dec ranges between negative 90 to 90 degrees.

4. Implementation

4.1 Algorithms

4.1.1 Greedy Search - Nearest Neighbor Heuristic

Greedy Search begins at a starting city and selects the nearest unvisited city as the next destination, repeating this process until all cities have been visited. It is computationally efficient with a time complexity of $O(n^2)$ but often produces sub-optimal routes due to its myopic nature. Here, it is used for benchmark purposes.

4.1.2 Ant System

AS simulates the foraging behavior of ants where virtual pheromone trails guide the search for optimal or near-optimal solutions. It efficiently explores the solution space and refines its search based on the quality of previous routes.

Given a set of cities represented as points in a Euclidean space, the goal is to find the shortest possible route that visits each city exactly once and returns to the origin city.

Let:

- C be the set of cities.
- n be the number of cities.
- $d(i, j)$ be the Euclidean distance between city i and city j.
- τ_{ij} be the pheromone level on edge (i, j).
- $\eta_{ij} = \frac{1}{d(i,j)}$ be the heuristic information (inverse of distance).

A solution (tour) can be represented as a permutation of cities

$$S = (c_1, c_2, \ldots, c_n)$$

where $c_i \in C$.

Roulette wheel selection tries to find optimal solutions by effectively balancing exploration and exploitation. It allows ants to make informed decisions based on both past performance (pheromones) and immediate conditions (heuristics) leading to improved convergence towards optimal solutions over iterations. The probability distribution $P(c_j|S_k)$ of moving from city c_i to city c_j, given the current solution S_k, is defined as

$$P(c_j|S_k) = \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{l \in N_k} (\tau_{il}^{\alpha} \cdot \eta_{il}^{\beta})}$$

where

1. N_k is the set of unvisited cities from city c_i.
2. α controls the influence of pheromone.
3. β controls the influence of heuristic information.

After all ants have constructed their tours, pheromones are updated based on the quality of solutions. Pheromones evaporate over time and are deposited based on tour quality. The update rule is given by

$$\tau_{ij}^{new} = (1-\rho)\tau_{ij}^{old} + \sum_{\text{all ants}} \left(\frac{Q}{L_k}\right)$$

where

1. L_k is the length of tour k
2. $0 < \rho < 1$ is the evaporation rate.
3. $Q > 0$ is a constant.
4. L is the length of the tour constructed by an ant.

4.1.3 Ant Colony Optimization Elite - ASElite

ASElite is a modified variant of AS where some ants are labeled as elite. These elite ants contribute more significantly to pheromone updates than others. The pheromone levels evaporate over time just like in traditional AS. However, pheromones are deposited based on the quality of all ants' solutions.

The pheromone update rule in ASElite is given by

$$\tau_{ij}^{new} = (1-\rho)\tau_{ij}^{old} + \sum_{\text{all ants}} \left(\frac{Q}{L_k}\right) + num\_elite \cdot \left(\frac{Q}{L_e}\right)$$

where

1. L_e is the length of the tour constructed by elite ants.
2. num_elite represents the number of elite ants.

4.1.4 Max-Min Ant System - MMAS

In MMAS, pheromone values are constrained within a range - between tau_min and tau_max. This prevents excessive pheromone accumulation on paths which can lead to premature convergence on sub-optimal solutions. Also, MMAS updates pheromones based solely on the best solution found in each iteration thus reinforcing high-quality paths more effectively.

After evaporation, pheromones are updated based on both the best solution and the global best solution
For each edge (i, j) in the best solution

$$\tau_{\text{best}_{ij}} \leftarrow \tau_{\text{best}_{ij}} + \frac{1}{d_{\text{best}}}$$

For each edge (i, j) in the global best solution

$$\tau_{\text{global}_{ij}} \leftarrow \tau_{\text{global}_{ij}} + \frac{1}{d_{\text{global}}}$$

where

1. τ_{min}: the minimum pheromone level.
2. τ_{max}: the maximum pheromone level, respectively.
3. d_{best}: the distance of the best solution found in the current iteration.
4. d_{global}: the distance of the global best solution found so far.

After updating, it is ensured that pheromone levels remain within bounds

$$\tau_{ij} = \begin{cases} \tau_{\text{min}}, & \text{if } \tau_{ij} < \tau_{\text{min}} \\ \tau_{\text{max}}, & \text{if } \tau_{ij} > \tau_{\text{max}} \\ \tau_{ij}, & \text{otherwise} \end{cases}$$

Every 10 iterations, a perturbation is applied to encourage exploration:

$$\tau_{ij} = (1 + U(-0.1, 0.1)) * \tau_{ij}$$

where $U(-0.1, 0.1)$ is a uniform random variable between -0.1 and 0.1.

This encourages exploration of the solution space and prevents the ants from becoming too biased towards paths that were succesful in the past.

4.2 Parameters

Initially, the number of ants is set to be proportional to the number of cities. The number of iterations is tweaked depending on the results of the graph (when it starts to converge). The evaporation rate, ρ is given a value of 0.5 to balance between retaining past knowledge and encouraging exploration. The pheromone influence, α is set to 1 to allow ants to exploit existing solutions while still exploring new ones. The heuristic influence, β is set to 2 to allow ants to give more weight to heuristic information and explore more optimal local choices but also to not rely too much on pheromone trails.

4.2.1 Results, n

	Greedy		AS		ASElite		MMAS	
n	Best distance	Time	Best distance	Time	Best distance	Time	Best distance	Time
50	2576	0.010	2444	35.57	2320	36.27	2423	35.40
100	2347	0.019	2166	62.01	2159	83.43	2247	64.21
500	5169	0.102	5060	351.74	5180	1240.53	4754	374.39

4.3 Hyperparameter Tuning using Bayesian Optimization

Using `gp_minimize` from `skopt`, Bayesian Optimization is used to find the best parameters (n_{ants}, ρ, α, β, τ_{min}, τ_{max}).
The search space for the hyperparameters is defined. Then, using the best parameters found, the MMAS algorithm is tested.

```
space = [
    Integer(50, 100),        # n_ants range
    Real(0.5, 1.0),          # rho range
    Real(1.0, 2.0),          # alpha range
    Real(1.0, 5.0),          # beta range
    Real(0.0, 0.5),          # tau_min range
    Real(5.0, 15.0)          # tau_max range
]
```

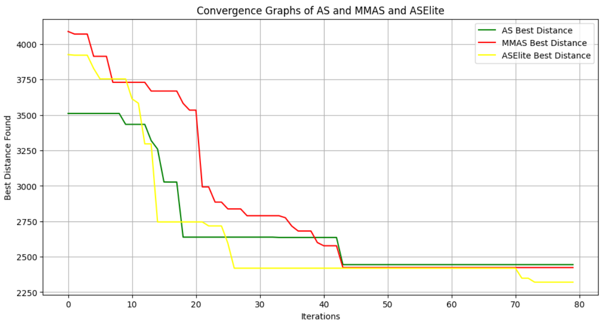
For the dataset of 50 cities (See results above), the Bayesian optimization process explores the defined search space and iteratively improves the parameter values to minimize the total distance traveled by the ants.

The best parameters found are: n_ants: 69, rho: 0.83, alpha: 1.93, beta: 4.38, tau_min: 0.15, tau_max: 10.24

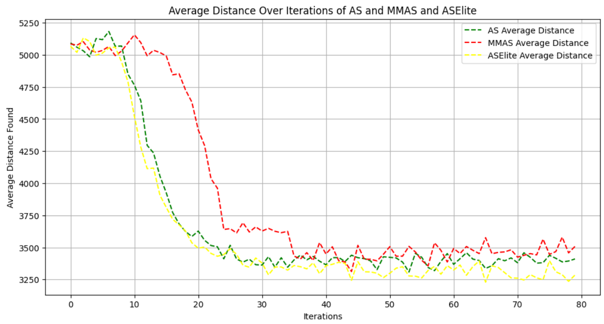
MMAS is run with the optimized hyperparameters. The minimum distance is now 1689. The results show that the shortest distance achieved with the tuned MMAS is significantly shorter than the distances produced by the untuned versions of Ant System, ASElite and MMAS.

4.3.1 Exploitation v/s Exploration

Exploration (searching the space for diverse solutions) and exploitation (focusing on improving the best solution) in the optimization process need to be balanced to avoid premature convergence (too much exploitation) or excessive randomness (too much exploration). The convergence graph tracks the best distance found to show how the algorithm narrows down on better solutions over time. The average distances graph shows the average quality of all solutions generated by the algorithm in each iteration.



(a)



(b)

Figure 1: (a) Convergence, (b) Average Distances

In AS, ants explore the solution space by using probabilities to select paths based on pheromone levels and heuristic information. The pheromone update involves contributions from all ants which encourages exploration of different paths. This is what leads to a slower convergence rate as sub-optimal paths may still receive reinforcement. Through the evaporation of pheromones, there is no accumulation on less favorable paths over time.

In ASElite, the elite ants prioritize some solutions. They contribute more significantly to the pheromone update process. This increases exploitation of high-quality paths while still allowing for exploration through other ants. The search follows promising areas of the solution space more quickly leading to faster convergence on optimal solutions.

MMAS improves the exploration-exploitation balance by implementing pheromone bounds (tau_min and tau_max). This prevents excessive pheromone accumulation on any single path and exploration is encouraged even when certain paths are performing well. MMAS updates pheromones based on the best solution found in each iteration. This reinforces high-quality paths while limiting the influence of sub-optimal paths. Periodic perturbations in pheromone levels help maintain diversity in the search process to prevent premature convergence.

5. Challenges

1. Recreating the galaxy into a graph
   For the problem to fit more into real-life scenario, I first attempted to map the data gathered about the position of the celestial bodies into an x, y, z 3-D Cartesian plane. I did so by converting the RA *(x-coord)* and Dec *(y-coord)* from degrees to radians and fixing the distance *(z-coord)* to 65 lightyears to scale the coordinates uniformly.
   Soln. - For the sake of simplicity and turning the focus to the algorithm, I opted to use a 2-D Cartesian plane to map the RA and Dec, like did Humphries (2023) who argued that it serves as a good approximation of the problem. Incorporating a 3-D plane was on the verge of becoming heavily reliant on physics, requiring to factor in other constraints, like angular magnitude of telescope slewing etc.

2. Max-Min AS Early Convergence
   Ants were following the same suboptimal path and converging too early.
   Soln. - Resetting the pheromones periodically every few iterations helped avoid stagnation by escaping local optima. The algorithm then forgot suboptimal paths and searched for better solutions.

3. Hyperparameter Tuning
   For a large number of cities, grid search was computationally expensive and time-consuming
   Soln. - Bayesian Optimization using *scikit-optimize*'s *gp_minimize* function was used to tune the parameters. It uses a probabilistic model to predict where good hyperparameters might be located to converge to optimal values more quickly than grid search which evaluates every combination without knowledge of past information.

4. τ values influence on MMAS Exploration-Exploitation
   The MMAS initially produced short distances at the start of the iterations because the pheromone matrix was initialized with maximum tau values. This led to an imbalance in the pheromone levels.
   Soln. - I printed the pheromone matrix at each iteration to visualize its values. I discovered that the difference between the minimum and maximum values in the pheromone matrix was too small. This small range caused the probability distribution to become too uniform resulting in excessive exploitation of certain paths. Then, I adjusted the $\tau_{\min}$ value to 0.00005 which helped balance between exploration and exploitation.

6. Conclusions

Three Ant Colony Optimization algorithm strategies were implemented: Ant System, ASElite, and Max Min Ant System. Each was tested with different parameter configurations. Bayesian Optimization was applied to identify the optimal parameters, which significantly improved the best solution found. The balance between exploitation and exploration for each strategy was analyzed and compared. It is still uncertain whether ACO would be effective for observing the galaxy which involves datasets on the order of thousands of points as very large datasets could not be tested on the Google Colab CPU. However, since ACO tends to scale well, it could be assumed to be a viable optimization strategy, especially when combined with a divide-and-conquer approach.

References

Beichman, C. (2014). Observations of transiting exoplanets with the james webb space telescope (jwst). *Publications of the Astronomical Society of the Pacific*, 126(946).

Carlson, S. (1997). Algorithm of the gods. *Scientific American Magazine*, 276(3).

Egemen Kolemen, N. J. K. (2007). Optimal trajectory control of an occulter-based planet-finding telescope. *Princeton University, Princeton, NJ.*

Fei Long, F. S. (2006). An improved ant colony algorithm and its application in routing computation of satellite networks. *IEEE The Proceedings of the Multiconference on Computational Engineering in Systems Applications.*

Humphries, M. (2023). *Astro-tsp: Traveling salesman problem based solutions for scheduling astronomical based observations* [Doctoral dissertation, California State Polytechnic University, Pomona].

Iddo Drori, B. K. (2020). Galaxytsp: A new billion-node benchmark for tsp. *MIT, Department of Electrical Engineering and Computer Science, Columbia University, Department of Computer Science.*

Rania Hassan, O. d. W., Babak Cohanim. (2005). A comparison of particle swarm optimization and the genetic algorithm. *46th AIAA Structures, Structural Dynamics, Materials Conference, Massachusetts Institute of Technology, Cambridge*.