Optimizing UAV-Based Delivery Networks via Graph Convolutional Networks and Ant Colony Optimization

Van Chau Huy, Hung Do Hoang Duy

***Abstract*—This paper addresses the problem of optimizing UAV-based delivery networks to maximize user satisfaction while minimizing flight cost. We explore two complementary approaches, Graph Convolutional Networks (GCNs) and Ant Colony Optimization (ACO). GCNs model the spatial and demand-based relationships within the delivery environment, enabling data-driven cluster-to-UAV assignments. Meanwhile, ACO employs decentralized, nature-inspired heuristics to derive effective routing strategies under real-world constraints. Through extensive experimentation, we compare both methods in terms of delivery efficiency, user coverage, and scalability.**

I. Introduction

Unmanned Aerial Vehicles (UAVs) play a critical role in time-sensitive delivery tasks, especially during emergencies. We investigate a problem where multiple UAVs are dispatched from distributed depots to deliver vital supplies, including medicine, equipment, and necessities, to geographically distributed user groups.

Each cluster has unique demand profiles and item priorities, with items assigned weight and importance scores. Our goal is twofold: (1) to determine the item allocations per UAV to maximize overall satisfaction, and (2) to identify routes that minimize total UAV travel while satisfying delivery time constraints.

To address this, we examine two distinct frameworks

- **Graph Convolutional Networks (GCNs)** Treating UAVs and clusters as nodes, GCNs capture topological relationships and enable learned cluster-to-UAV assignments.
- **Ant Colony Optimization (ACO)** A probabilistic, nature-inspired algorithm that generates UAV paths via pheromone and heuristic value updates to maximize user coverage.

We compare both approaches under realistic assumptions, highlighting their respective trade-offs in performance and scalability.

II. System Formulation

A. UAV Maximize Capacity Formulation

For each UAV m, let $H_m \subseteq \{1, \dots, N\}$ be the set of item-types it carries. We require

$$\sum_{n \in H_m} w_n \le w_m,$$

where w_n is the weight of one unit of item n, and w_m is UAV m's maximum payload.

B. Minimize Path Weight Formulation

Let $C_m \subseteq \{1, \dots, C\}$ be the clusters served by UAV m, and $\mathrm{dist}(m,i)$ the Euclidean distance from UAV m's depot to cluster i. We seek to minimize total flight cost

$$\min \sum_{m=1}^{M} \sum_{i \in C_m} \mathrm{dist}(m,i).$$

C. Maximize User Satisfaction Formulation

Each cluster i has C_i users requesting item n_i, with importance score u_{n_i}. Introduce binary $\delta_{mi} = 1$ if UAV m serves cluster i. Then

$$\max \sum_{m=1}^{M} \sum_{i=1}^{C} \delta_{mi} \left(C_i \cdot u_{n_i}\right).$$

Optionally combine with a distance penalty

$$\max \sum_{m,i} \delta_{mi} \left(C_i u_{n_i}\right) - \lambda \sum_{m,i} \delta_{mi}\, \mathrm{dist}(m,i).$$

III. Graph Convolutional Network (GCN) Approach

A. System Model and Parameters

- N: numbers of items, each items weight w_n each items importance (bias) u_n.
- C: number of clusters, each cluster i has C_i people and demands item t_i.
- M: number of UAVs, UAV m located at position (x_m, y_m, z_m) and has maximum capacity w_m.

B. Graph Construction

Define a directed graph $G = (V, E)$

$$V = \{u_1, \dots, u_M\} \cup \{c_1, \dots, c_C\}$$

where each UAV u_m and each cluster c_i are nodes. We create an edge $(u_m, c_i) \in E$ for every potential service link.

Node features

Each UAV node u_m is endowed with

$$\mathbf{x}_{u_m} = \left[x_m,\, y_m,\, z_m,\, w_m,\, 0,\, 0\right]^\top \in \mathbb{R}^6,$$

and each cluster node c_i with

$$\mathbf{x}_{c_i} = \left[x_i,\, y_i,\, 0,\, 0,\, C_i,\, n_i\right]^\top \in \mathbb{R}^6,$$

where w_m is UAV payload, C_i user count, and n_i item index.

Edge features

For each (u_m, c_i), let

$$d_{mi} = \| (x_m, y_m, z_m) - (x_i, y_i, 0) \|$$

$$t_{mi} = \frac{d_{mi}}{v}, \quad \omega_{mi} = C_i \, w_{n_i}, \quad u_{mi} = u_{n_i},$$

and set $\mathbf{e}_{mi} = [\, d_{mi},\, t_{mi},\, \omega_{mi},\, u_{mi} \,]^\top \in \mathbb{R}^4$.

C. GCN Architecture

We employ a two-layer GCN followed by a linear read-out on UAV nodes

$$\mathbf{H}^{(1)} = \sigma\big(\hat{A}\,\mathbf{X}\,W^{(1)}\big), \quad \mathbf{H}^{(2)} = \sigma\big(\hat{A}\,\mathbf{H}^{(1)}\,W^{(2)}\big),$$
$$\mathbf{Z} = \mathbf{H}^{(2)}\,W^{(o)}, \qquad \widehat{Y}_m = \mathrm{sigmoid}\big(\mathbf{Z}_{u_m}\big) \in \mathbb{R}^C.$$

Here, $\mathbf{X} \in \mathbb{R}^{(M+C)\times 6}$ denotes the node feature matrix, where each row represents either a UAV or a cluster node. The matrix $\hat{A}$ is the symmetrically normalized adjacency matrix with self-loops, used to ensure stable message propagation in the GCN layers. The trainable weight matrices are $W^{(1)}, W^{(2)} \in \mathbb{R}^{6\times H}$ for the hidden layers, and $W^{(o)} \in \mathbb{R}^{H\times C}$ for the output layer, where H is the hidden dimension. The activation function σ is the sigmoid function, chosen for its smooth and bounded output. Finally, only the first M rows of the output matrix $\mathbf{Z}$, corresponding to the UAV nodes $\mathbf{Z}_{u_1}, \ldots, \mathbf{Z}_{u_M}$, are used to compute the predicted cluster assignment probabilities.

D. Training and Inference

We frame the problem as a multi-class classification over C clusters for each UAV. Let $y_m \in \{1, \ldots, C\}$ be the ground-truth best cluster for UAV m. We minimize the cross-entropy loss

$$\mathcal{L} \;=\; -\sum_{m=1}^{M} \log\big[\widehat{Y}_m\,[y_m]\big].$$

We train with Adam (learning rate 10^{-3}), batch size 1 (full graph), for 200 epochs.

At inference, each UAV m is assigned to cluster $\arg\max_i \widehat{Y}_m[i]$, producing a near-optimal assignment that maximizes $C_i\,u_{n_i}$ while implicitly accounting for distance through the learned embeddings.

This GCN framework thus learns from data to solve the maximization problem in a single forward pass, offering scalability and adaptability beyond hand-crafted heuristics.

E. Training Evaluation

Change the activation function of GCN model from relu into sigmoid makes model works better. While learning rate equal 1e-3 is still the best. Sigmoid provides smooth, differentiable curves that may help small or shallow networks learn more stable gradient updates than ReLu.

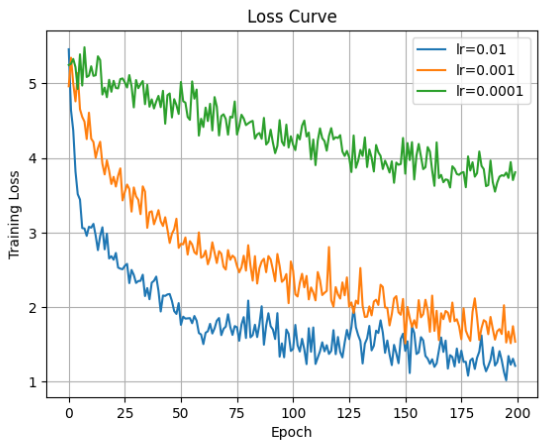

Fig. 1: Training Loss Visualization

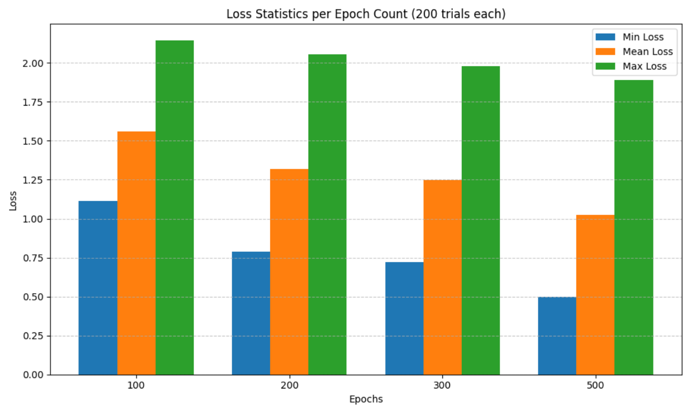

Fig. 2: Comparing Epochs with Learning Rate 1e−2

F. Epoch Evaluation

Figure 2 shows that increasing the number of epochs results in lower mean and maximum loss, indicating improved model convergence. The minimum loss also decreases slightly, while the gap between max and min loss narrows, reflecting more consistent training. Overall, the chart highlights that longer training leads to better and more stable performance.

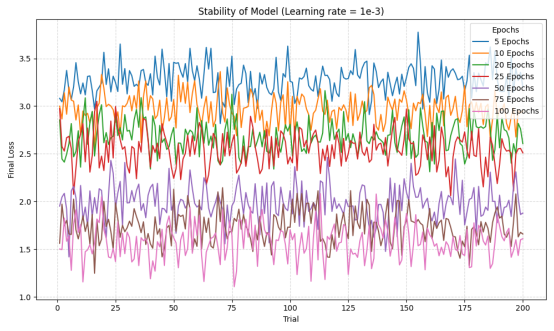

Fig. 3: Stability Across Epochs (Learning Rate 1e−2)

Figure 3 illustrates the final loss distribution across 200 trials for various epoch settings. It is evident that as the number of epochs increases, the average loss tends to decrease and the fluctuation becomes smaller. In particular, settings with 75 and 100 epochs show significantly lower and more stable loss compared to those with 5 or 10 epochs. This suggests that longer training leads to better convergence and more consistent model performance.

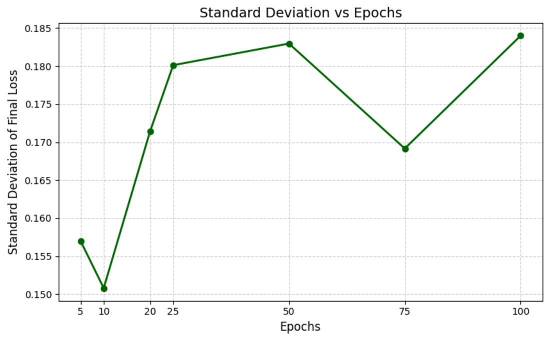

Fig. 4: Standard Deviation of Final Loss Across Epochs (Learning Rate 1e−2)

Figure 4 shows a fluctuating trend in the standard deviation of final loss as the number of epochs increases. The deviation is lowest at 10 epochs, suggesting more stable outcomes at that point. However, it increases significantly from 10 to 50 epochs and peaks at 100 epochs. This may indicate that training for too many epochs introduces variability, possibly due to overfitting or unstable training dynamics.

IV. Ant Colony Optimization (ACO) Approach

A. Problem Overview

The Ant Colony Optimization (ACO) approach aims to solve the delivery task described in Section IV by generating feasible routes for UAVs that maximize user satisfaction while minimizing travel cost. Each UAV must serve a subset of clusters under flight time and payload constraints.

Rather than reformulating the problem, we adopt the same objective and constraints defined in Section IV, specifically targeting both the satisfaction formulation

$$\max \sum_{m=1}^{M} \sum_{i=1}^{C} \delta_{mi} \left(C_i \cdot u_{n_i} \right).$$

and the combined satisfaction and path penalty optimization

$$\max \sum_{m,i} \delta_{mi} \left(C_i u_{n_i} \right) \; - \; \lambda \sum_{m,i} \delta_{mi} \, \mathrm{dist}(m,i).$$

B. Proposed Algorithm Design

The algorithm we propose employs a region-based ACO framework specifically designed for heterogeneous UAV routing. The key idea is to exploit local and neighboring cluster assignments while ensuring that each UAV remains within regional limits and satisfies flight time constraints.

Initially, the positions of the UAVs $\{\mathbf{s}_m\}_{m=1}^M$ and the coordinates of the cluster $\{\mathbf{c}_i\}_{i=1}^C$ are grouped, defining the regional centers $\{\mu_r\}_{r=1}^M$. For each region r, a radius R_r is calculated to ensure that at least $\left\lfloor \frac{C}{M} \right\rfloor$ clusters are enclosed.

$$R_r = \min \left\{ r \,|\, \{ i \| \mathbf{c}_i - \mu_r \| \le r \} | \ge \left\lfloor \frac{C}{M} \right\rfloor \right\}.$$

Neighboring region sets $\mathcal{N}_r$ are identified via centroid proximity to allow inter-regional transitions when needed.

Pheromone and heuristic matrices are initialized as

$$\tau_{mi}^{(0)} = \frac{1}{\|\mathbf{s}_m - \mathbf{c}_i\| + \varepsilon}, \quad \eta_{mi} = \frac{C_i}{\|\mathbf{s}_m - \mathbf{c}_i\|}.$$

At each iteration, each UAV constructs its path based on a probability distribution over eligible clusters, balancing exploration and exploitation

$$P_{mi} = \frac{\left(\tau_{mi}^{(t)}\right)^{\alpha} \left(\eta_{mi}\right)^{\beta}}{\sum_{k \in \mathcal{F}_m^t} \left(\tau_{mk}^{(t)}\right)^{\alpha} \left(\eta_{mk}\right)^{\beta}},$$

where α and β are tunable parameters, and $\mathcal{F}_m^t$ is the set of feasible clusters for UAV m at iteration t.

Each UAV continues to select clusters until flight time and regional restrictions are violated. A candidate solution consists of all paths $\{\mathcal{R}_m\}_{m=1}^M$ built this way.

After route construction, pheromone levels are updated as

$$\tau_{mi}^{(t+1)} = (1-\rho) \cdot \tau_{mi}^{(t)} + \Delta\tau_{mi}^{\text{best}},$$

where ρ is the rate of evaporation, and the best solution among the ants contributes

$$\Delta\tau_{mi}^{\text{best}} = \begin{cases} \lambda / T_m, & \text{if } i \in \mathcal{R}_m^{\text{best}}, \\ 0, & \text{otherwise.} \end{cases}$$

This iterative process continues until convergence or a maximum iteration count is reached. The final solution $\{\mathcal{R}_m\}_{m=1}^M$ maximizes both satisfaction score and the net satisfaction with penalty subject to UAV time and coverage constraints.

C. Numerical Evaluation

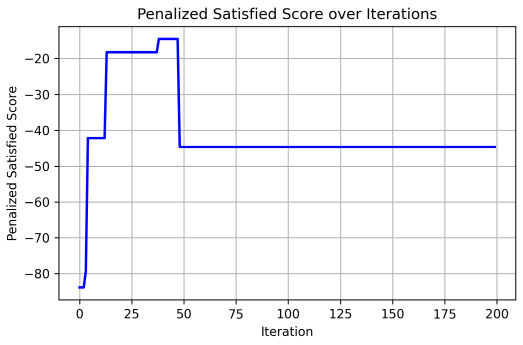

Fig. 5: Penalized satisfaction score over iterations.

To evaluate the performance of the proposed UAV routing approach based on ACO, we conducted extensive experiments

Algorithm 1 ACO-based UAV Routing under Regional and Time Constraints

```
1: Inputs $M$, $C$, $T^{\text{uav}}_{\max}$ $\bar{T}_{\max}$, $\{\mathbf{s}_m\}_{m=1}^{M}$, $\{\mathbf{c}_i\}_{i=1}^{C}$, $\{C_i\}_{i=1}^{C}$, $\{u_{n_i}\}$.
2: Assign $C$ clusters into $M$ regions $\{\mathcal{C}_r\}_{r=1}^{M}$ with centroids $\{\boldsymbol{\mu}_r\}$ and neighborhoods $\mathcal{N}_r$
3: Initialize pheromone matrix $\tau_{mi}$ and heuristic matrix $\eta_{mi}$ $\eta_{mi} = \frac{C_i}{\|\mathbf{s}_m - \mathbf{c}_i\|}$,  $\tau^{(0)}_{mi} = \frac{1}{\|\mathbf{s}_m - \mathbf{c}_i\| + \varepsilon}$
4: for iteration $t = 1$ to $N_{\max}$ do
5:   Update dynamic parameters $\alpha_t$, $\beta_t$
6:   for each ant $a = 1$ to $N_a$ do
7:     Initialize positions $\mathbf{x}_m \leftarrow \mathbf{s}_m$, time budget $T_m \leftarrow T^{\text{uav}}_{\max}$, visited set $\mathcal{V} \leftarrow \emptyset$
8:     while $T_m > 0$ do
9:       for each UAV $m = 1$ to $M$ do
10:        Define candidate set $\mathcal{F}_m \leftarrow \mathcal{C}^{(a)}_r \cup \bigcup_{r' \in \mathcal{N}_r} \mathcal{C}^{(a)}_{r'}$ prune by $T^{\text{travel}}_{m,i} \le T_m$,  $\bar{T}^{(i)}_m \le \bar{T}_{\max}$
11:        Compute probability
               $$P_{mi} = \frac{(\tau_{mi})^{\alpha_t} (\eta_{mi})^{\beta_t}}{\sum_{k \in \mathcal{F}_m} (\tau_{mk})^{\alpha_t} (\eta_{mk})^{\beta_t}}$$
12:        Select cluster $i^* \in \mathcal{F}_m$ by $P_{mi}$
13:        Update$\mathbf{x}_m \leftarrow \mathbf{c}_{i^*}$, $\mathcal{V} \leftarrow \mathcal{V} \cup \{i^*\}$
14:        Update $T_m \leftarrow T_m - T^{\text{travel}}_{m,i^*}$
15:      end for
16:    end while
17:    Evaluate score: $S_a = \sum_{i \in \mathcal{V}} C_i u_{n_i}$, $T^{\text{avg}}_a = \frac{1}{M} \sum_{m=1}^{M} (T^{\text{uav}}_{\max} - T_m)$, $S^{\text{penal}}_a = S_a - \lambda \sum_{m=1}^{M} (T^{\text{uav}}_{\max} - T_m)$
18:    Update global best if $S_a$ or $S^{\text{penal}}_a$ improves
19:  end for
20:  Update pheromone:
               $$\tau_{mi} \leftarrow (1-\rho)\tau_{mi} + \frac{\lambda}{T^{\text{avg}}_a + \epsilon}, \quad \forall (m,i) \in \mathcal{R}^*_m$$
21: end for
22: Output Optimal routes $\{\mathcal{R}^*_m\}_{m=1}^{M}$ maximizing total satisfaction under time and region constraints
```

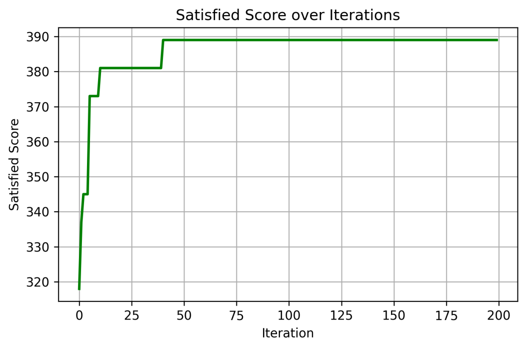

Fig. 6: Satisfaction score over iterations.

and analyzed key metrics in the iterative optimization process. The convergence behavior and the quality of the solution are illustrated in Figures 5, 6, 7, 8, and 9

Figures 5 and Figure 6 present the evolution of the penalized satisfaction score and the raw satisfaction score, respectively, over iterations. We observe that both scores rapidly improve during the initial iterations and subsequently stabilize, indicating convergence. The penalized satisfaction score initially starts at a low value (approximately -72), increases sharply, and stabilizes around -45, reflecting a balance between user satisfaction and route time penalties. The rapid drop near the 50^{th} iteration in the penalized satisfaction score was due to the algorithm's prioritising service people over service time. Similarly, the raw satisfaction score rises from around 365 to approximately 389, demonstrating the algorithm's effectiveness in maximizing service coverage. The huge plummet in penalized score was taken for the occurrence of the small rise near the 50^{th} iter.

Figures 7 , 8, and 9 illustrate the auxiliary metrics that are implicitly considered in the calculation of the satisfaction score, including average flight time, total flight time, and number of people helped.

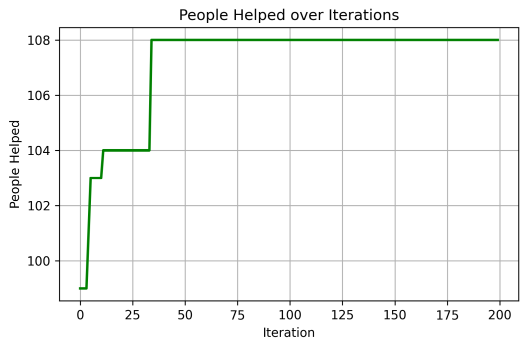

Fig. 7: Number of people helped over iterations.

Figure 7 depicts the number of people assisted as the optimization progresses. This metric increases from an initial value near 99 to over 108, illustrating the algorithm's ability to enhance people's reach relative to each's cluster importance score while respecting energy and time constraints, converging at 108.

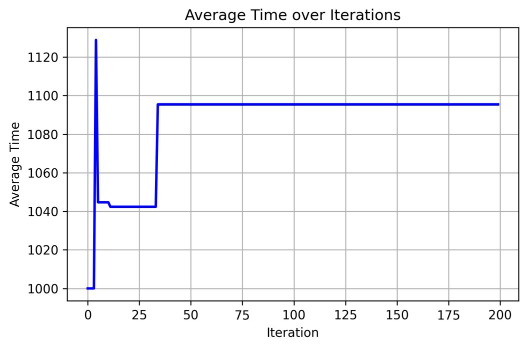

Fig. 8: Average UAV flight time over iterations.

Figure 8 shows the average UAV flight time across iterations. We note an initial fluctuation due to exploration and route adjustments, which found that better people helped value, meaning a better satisfaction score. Then, it is followed by a gradual convergence to around 1090 seconds, suggesting efficient time allocation among UAVs, which resulted in a better performance regarding penalized satisfaction.

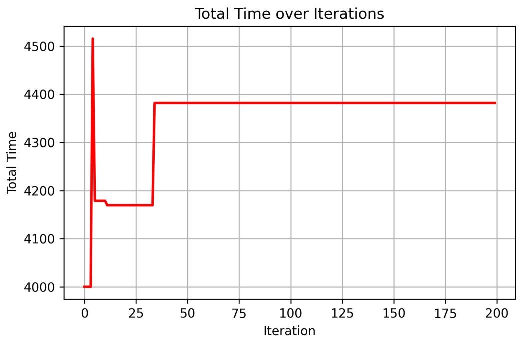

Fig. 9: Total UAV flight time over iterations.

Finally, figure 9 displays the total time utilized by all UAVs combined. Similar to the average time trend, we observe early fluctuations before stabilizing and converging near 4380 seconds, indicating effective global time optimization.

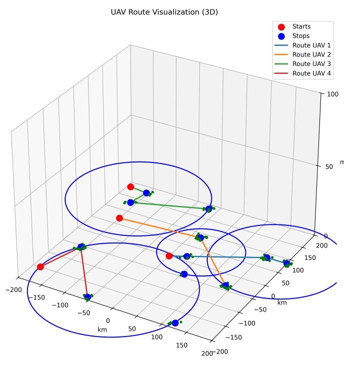

Fig. 10: Total UAV flight time over iterations.

Figure 10 visualizes final UAV routes in 3D. Distinct colored paths and clear separations among UAVs demonstrate effective task partitioning and route feasibility. The smooth, non-overlapping trajectories confirm the algorithm's ability to generate practical, collision-free paths, validating its suitability for real-world UAV fleet routing and mission planning applications.

Overall, these numerical results confirm that the proposed approach effectively balances and converges user satisfaction, flight time constraints, and service coverage, leading to a stable and high-quality solution after sufficient iterations.

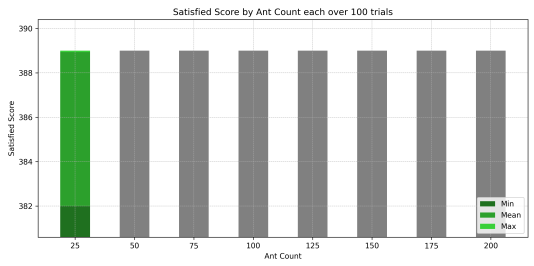

Fig. 11: Satisfaction score performance

D. Performance Evaluation

Figure 11 shows satisfaction scores over 100 trials, which remain consistently high and stable across different ant counts. At lower counts (25), there is a noticeable spread between minimum and maximum values, but suggesting possible outliers since the mean stays close to the maximum. As the ant count increases beyond 50, penalized satisfaction scores converge tightly, with min, mean, and max becoming nearly identical. This demonstrates that while small colonies can achieve similar raw satisfaction levels, larger colonies significantly improve solution robustness and feasibility.

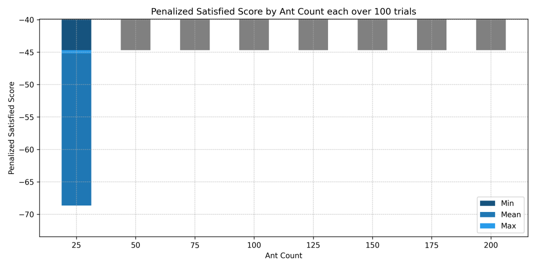

Fig. 12: Penalized satisfaction score over ant counts.

Figure 12 shows penalized satisfaction scores over 100 trials per ant count. Similar to the unpenalized case, lower ant counts exhibit early degradation, especially in the maximum value, likely due to outliers, as the mean remains close to the minimum. At higher counts, scores converge tightly with virtually no variance throughout all 100 trials Nonetheless, across all ant counts, the algorithm is still able to find optimal solutions among the trials, indicating that even small colonies can achieve high satisfaction, while larger colonies further improve the reliability by reducing penalty risks.

Figure 13 shows the algorithm's runtime as a function of ant count. As expected, computational time increases steadily from 25 to 200 ants due to the linear scaling of solution construction and pheromone updates. While larger colonies can improve solution quality, this comes at a clear cost: average runtimes rise from about 6 seconds (25 ants) to nearly 48 seconds (200 ants). This highlights the need to balance solution quality and computational efficiency in practical ACO applications.

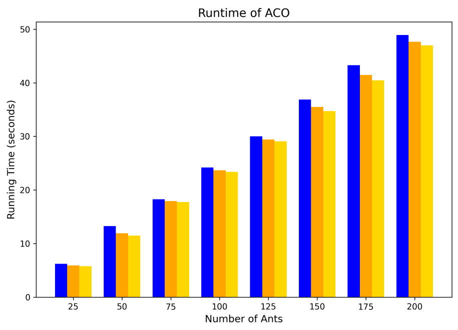

Fig. 13: Runtime of ACO by ant count.

References

[1] A. Momenzadeh and M. F. Sabahi, “Throughput-Oriented IRS Beamforming in 5G Networks via Graph Convolutional Networks,” *arXiv preprint arXiv:2505.04894*, May 2025. [Online]. Available: https://arxiv.org/pdf/2505.04894

[2] T. N. Kipf and M. Welling, “Semi-Supervised Classification with Graph Convolutional Networks,” in *Proc. Int. Conf. Learn. Representations (ICLR)*, Toulon, France, Apr. 2017. [Online]. Available: https://arxiv.org/abs/1609.02907

[3] T. Wang, J. Wu, X. Guo, Y. Li, and G. Long, “Graph Convolutional Collaborative Filtering for Recommendation,” *arXiv preprint arXiv:2503.03313*, Mar. 2025. [Online]. Available: https://www.alphaxiv.org/abs/2503.03313

[4] K. Geirhos, D. Sundararajan, and T. Schirrmeister, “Generalizing Convolutional Neural Networks to Non-Euclidean Domains in Computer Vision,” *arXiv preprint arXiv:2407.21525*, Jul. 2024. [Online]. Available: https://arxiv.org/abs/2407.21525

[5] S. Vashishth, P. Saini, and P. Talukdar, “Graph-Based Models for Natural Language Processing: A Survey,” *arXiv preprint arXiv:2209.15003*, Sep. 2022. [Online]. Available: https://arxiv.org/abs/2209.15003

[6] M. Dorigo, M. Birattari, and T. Stützle, ”Ant colony optimization,” *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28–39, Nov. 2006, doi: 10.1109/MCI.2006.329691.

[7] M. Dorigo and L. M. Gambardella, ”Ant colony system: a cooperative learning approach to the traveling salesman problem,” *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 53–66, Apr. 1997, doi: 10.1109/4235.585892.