2023 IEEE Symposium Series on Computational Intelligence (SSCI)
Mexico City, Mexico. December 5-8, 2023

Graph Convolutional Network based Ant Colony Optimization for Robot Task Allocation

Jiang Qiu*, Yi Liu†, Yilan Yu*, Wei Li†
Academy for Engineering and Technology
Fudan University, Shanghai, China
*Email: {qiuj21, ylyu22}@m.fudan.edu.cn
† Email: {liuyi_, fd_liwei}@fudan.edu.cn

***Abstract*—The robot task allocation is a crucial problem in logistics and distribution where robots are required to perform an array of tasks, with differing locations and numbers. As a result, optimally allocating tasks to available robots to minimize associated costs has become a challenging but essential optimization problem. This paper presents a Graph Convolutional Network (GCN) based Ant Colony Optimization (ACO) denoted as GCN-ACO, to solve the robot task allocation problems, which are formulated as the Travelling Salesman Problems (TSP) and Vehicle Routing Problems (VRP). The GCN-ACO algorithm comprises two stages. In the first stage, a GCN model is trained to predict a heatmap, which represents the probability of each edge belonging to the optimal route within the graph. In the second stage, we integrate the predicted heatmap into ACO to guide the ant colony to select edges with greater potential during the search process. We evaluate our approach's performance through testing on standard TSP and VRP datasets. The experimental results demonstrate that our proposed method has a faster convergence rate and a higher quality of solutions compared to the baseline approaches.**

***Index Terms*—Graph Convolutional Network (GCN), Ant Colony Optimization (ACO), Robot Task Allocation**

I. INTRODUCTION

With the development of artificial intelligence technology, utilizing robots to execute tasks can better meet the demands of rapidly changing tasks, make appropriate adjustments in a short period of time, thereby ensuring execution efficiency and enhancing resource utilization [1]. In order to arrange the task execution sequence accurately, robots require efficient and reasonable task allocation during execution.

According to the number of robots, the robot task allocation problem can be divided into single-robot and multi-robot task allocation [2], which constitutes a challenging optimization problem. [3]. Mathematically speaking, a multi-task allocation problem for a single robot can be deemed identical to the Travelling Salesman Problem (TSP) [4]. TSP is a complex challenge that involves determining the shortest route for visiting a given set of destinations. Similarly, multi-task allocation for multiple robots can be modelled as a vehicle routing problem (VRP) where a team of robots is required to complete a set of tasks that are scattered throughout the environment and subsequently return to the depot in such a way that the total cost is minimized while also satisfying capacity constraints [5]. Such combinatorial optimization problems have been extensively studied. Researchers have employed exact methods such as mixed integer programming (MIP) [6], and heuristic algorithms like Lin–Kernighan–Helsgaun (LKH) [7], as well as meta-heuristic methods [8]. Due to the task allocation problem being an NP-hard problem, heuristic algorithms such as ant colony optimization (ACO) have advantages in terms of computational efficiency [9].

Ant colony optimization (ACO) [10] is a popular optimization technique inspired by the behavior of ant colonies in finding the shortest path between their nest and food source. It is a meta-heuristic algorithm that can be applied to solve various optimization problems, including task allocation problems [11]. The pheromone trails of ACO are used to guide the search process of ants towards the optimal solution. The initial uniform distribution of pheromones does not provide clear guidance, resulting in a blind search during the early stages and requiring numerous iterations to converge to an optimal or sub-optimal solution.

To address the aforementioned issues, this paper presents a new approach that integrates Ant Colony Optimization (ACO) and Graph Convolutional Network (GCN) [12] to solve the task allocation problem. GCN possesses the capability to learn the relationships among nodes and the topological structure of a graph to extract rich features for edge prediction. Hence, we employ the predicted probabilities from GCN in conjunction with pheromones as guidance for ant colony search. We evaluate our approach on a set of benchmark problems and compare it with existing algorithms. The results demonstrate that our method has significant advantages over traditional ACO in solving the task allocation problems.

The main contributions of this work are as follows:

- We proposes a hybrid ant colony optimization based on graph convolutional network (GCN-ACO) for quickly solving robot task allocation problems.
- GCN-ACO exhibits superior generalization, enabling fast discovery of optimal solutions even when task locations or quantities are modified, thus avoid searching from scratch.

This research was supported in part by Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103), and in part by Scientific Research Development Center in Higher Education Institutions by the Ministry of Education, China (No.2021ITA10013), and in part by the Shanghai Engineering Research Center of AI and Robotics, and in part by the Engineering Research Center of AI and Robotics, Ministry of Education, China.

- The experiments indicate that GCN-ACO proposed in this paper has a faster convergence speed and can detect higher quality solutions compared to traditional ant colony optimization algorithms and other baselines.

The structure of the remaining sections of this paper is outlined as follows. In Section II, we introduce a brief overview of related work. In Section III, we formulate the robot task allocation problems as TSP and VRP problems. Section IV provides a detailed description of the proposed GCN-based ACO algorithm. The experimental results are provided in Section V, and an elaborate discussion is presented on the effectiveness of the proposed methods. Finally, in Section VI, we conclude the paper and discuss future work.

II. RELATED WORK

The robot task allocation problem has been recognized to be an NP-hard problem, and has thus been extensively investigated by the academic community. In general, the task allocation problem can be modeled as a combinatorial optimization problem, such as TSP or VRP [13]. At present, there are numerous methods that can be used to solve this type of problem, including exact methods and heuristic algorithms. When using exact methods to solve combinatorial optimization problems, they are typically formulated as mixed-integer linear programs (MILPs), in which case branch-and-bound is the exact method of choice [14]. Although exact methods can provide the precise optimal solution, they need a significant amount of time. Conversely, heuristic algorithms can balance solution quality and computational efficiency, allowing for optimal or near-optimal solutions to be found in less time. Therefore, numerous heuristic methods have been proposed for solving task allocation problems, such as ant colony optimization [15], particle swarm optimization [16], genetic algorithms [17], and so on.

Ant colony optimization was initially developed to solve the classic travelling salesman problem, by taking inspiration from ants' ability to find the shortest path between food sources and their nests [18]. The algorithm gradually constructs feasible solutions using a probability model, and searches for the optimal solution through pheromones and positive feedback mechanisms. However, traditional ant colony algorithms often require multiple iterations to converge and are susceptible to getting trapped in local optima. Therefore, researchers have proposed several corresponding improved methods. For example, MAX-MIN Ant System (MMAS) [19], one of the most effective variants of ACO, restricts the value of pheromones within a certain range to balance the exploration-exploitation behavior of ants and prevent them from becoming stuck in local optima. In [20], an ant colony optimization with a warm-up process is proposed, whereby ant colony's iteration process is simulated based on the cost of the edges to initialize the pheromone matrix so as to achieve the same or even better results in fewer iterations. However, this method is still prone to falling into local optimal solutions.

In recent years, there has been an increasing interest in integrating machine learning techniques into meta-heuristic algorithms, such as ant colony optimization, to solve combinatorial optimization problems [8]. A parallel Kmeans-Elistist ACO approach was proposed in [21] to solve TSP/mTSP problems. The algorithm is performing parallel computations on a GPU for faster processing and comprises three stages: clustering the given dataset's points with K-means clustering, using elite ant colony optimization to find the shortest path in each cluster, connecting each cluster at the closest point to the other. [22] proposed an ML-ACO algorithm for solving the orienteering problems. The algorithm begins by mapping points from orienteering problem instances with known optimal solutions to a feature space. It then trains an SVM model to learn the decision boundary in the feature space for predicting the probability of edges in the graph belonging to the optimal path. Predicted probability values are then used as the initial pheromone matrix for the ant colony optimization to enhance its performance. However, using this method requires designing feature extraction formulas manually and different feature extraction methods can lead to vastly different results.

III. PRELIMINARY

A. Single Robot Task Allocation

In the scenario where there is only one robot operating in a given environment, the task allocation could be represented using the mathematical Travelling Salesman Problem (TSP) framework [2]. The problem can be described as follows: There is a robot R in the depot, which needs to perform a series of tasks $T = \{T_1, T_2, \ldots, T_n\}$. The depot and task points form a fully-connected weighted graph $G = (V, E)$, where V_0 indicates the location of the depot, V_i represents the position of the task $T_i (i = 1, 2, \ldots, n)$, $E = \{e_{ij} = (V_i, V_j) \mid 0 \leq i, j \leq n, i \neq j\}$, and the Euclidean distance of edge e_{ij} is $d_{ij} (i, j \in V, d_{ij} = d_{ji})$. The robot must travel from depot V_0 to all other task points, visiting each location only once and returning to the starting point. The objective is to find an optimal traversal route $V' = \{V'_0, V'_1, \ldots, V'_n, V'_{n+1}\}$ with $V'_0 = V_0$ and $V'_{n+1} = V_0$, which minimizes the total cost of the robot. The cost is given by Equation (1):

$$cost(V') = \sum_{i=0}^{n} d_{V'_i V'_{i+1}} \tag{1}$$

B. Multiple Robots Task Allocation

Considering a scenario in which multiple robots are located in a depot, with each robot required to complete a series of assigned sub-tasks while adhering to its maximum capacity limits. Such a scenario can be modelled as a vehicle routing problem (VRP), and its mathematical model can be described as follows [23]: the depot and each task are represented as vertices V in a fully-connected weighted graph $G = (V, E)$, where edges $E = \{e_{ij} = (V_i, V_j) \mid 0 \leq i, j \leq n, i \neq j\}$ represent connections between nodes. R represents the number of robots, $T = \{T_1, T_2, ..., T_n\}$ is the task set, and C is the maximum capacity of each robot. If e_{ij} is included in the route of robot r, $x^r_{i,j}$ equals 1, otherwise, $x^r_{i,j}$ equals 0. c^r_i represents

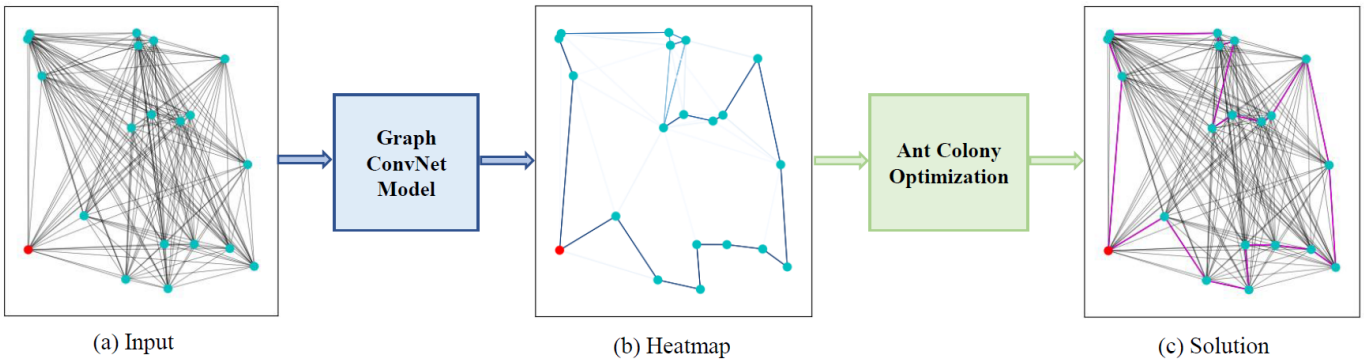


Fig. 1. Overview of GCN-ACO. Given a fully-connected graph as input (a), the GCN model outputs a heatmap (b) that represents the probability of each edge belonging to the optimal path. The ACO is then used to search for a solution (c) based on the heatmap and pheromone.

the remaining capacity of the robot r after completing task T_i, while m_j indicates the demand of task T_j. The objective is to minimize the total cost of all robots as illustrated in Formula (2), where d_{ij} represents the cost of e_{ij}.

$$\text{minimize} \sum_{r\in R}\sum_{\{i,j\}\in E} x_{i,j}^r d_{i,j} \tag{2}$$

$$\text{s.t.} \sum_{r\in R}\sum_{j\in V} x_{i,j}^r = 1, i \in T, j \neq i \tag{3}$$

$$\sum_{r\in R}\sum_{i\in V} x_{i,j}^r = 1, j \in T, i \neq j \tag{4}$$

$$\sum_{r\in R}\sum_{j\in T} x_{0,j}^r = R \tag{5}$$

$$\sum_{r\in R}\sum_{i\in T} x_{i,0}^r = R \tag{6}$$

$$c_j^r = \begin{cases} c_i^r - m_j, & \text{if } x_{i,j}^r = 1, \{i,j\} \in E, j \neq 0 \\ C, & \text{if } j = 0 \end{cases} \tag{7}$$

$$m_j \leq c_i^r \leq C, \quad \text{if } x_{i,j}^r = 1, i \in V, j \in T \tag{8}$$

$$x_{i,j}^r \in \{0,1\}, \{i,j\} \in E \tag{9}$$

Equations (3) and (4) indicate that each task can only be executed once. Constraints (5) and (6) ensure that each robot departs from and ultimately returns to the depot. Equation (7) demonstrates the update process of the remaining capacity of robot r. Constraint (8) ensures that the remaining capacity of the vehicle does not fall below the demand of the next task T_j and does not exceed the maximum capacity C of the robot r. Constraint (9) guarantees that $x_{i,j}^r$ is binary.

IV. Graph Convolutional Network based Ant Colony Optimization

In this section, we will introduce the basic ant colony optimization followed by proposing a novel ant colony optimization based on graph convolutional networks to address the robot task allocation problems.

A. Basic Ant Colony Optimization

The ant colony optimization was initially proposed to solve the TSP problem, in which the goal is to find the shortest route to link a series of cities. The Ant System applied to the TSP can be described as follows:

Suppose the number of ants is m, the number of cities is n, the distance between $city_i$ and $city_j$ is $d_{ij}(i, j = 1, 2, ..., n)$. The pheromone concentration on the connection path between $city_i$ and $city_j$ at time t is $\tau_{ij}(t)$. At the initial moment, m ants are randomly placed in n cities, and pheromone on each path is the same $\tau_{ij}(0) = \tau_0$. Then ant_k decides the next city to visit based on the pheromone and heuristic information along the path between cities. The probability of ant_k transferring from $city_i$ and $city_j$ at time t is shown in formula (10).

$$p_{ij}^k(t) = \begin{cases} \frac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}(t)]^\beta}{\sum_{s\in J_k}[\tau_{is}(t)]^\alpha \cdot [\eta_{is}(t)]^\beta}, & j \in J_k \\ 0, & other \end{cases} \tag{10}$$

In the formula (10), $\eta_{ij}(t) = \frac{1}{d_{ij}}$ is the heuristic function, expressed as the reciprocal of the distance between $city_i$ and $city_j$. J_k represents a set of cities to be visited for ant_k. α and β indicate the relative importance of the pheromone and the heuristic function respectively. When all ants have completed their search during one iteration, the pheromone on each path needs to be updated according to formula (11).

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^k, \quad 0 < \rho < 1 \tag{11}$$

where ρ represents pheromone evaporation coefficient along the path, $\Delta\tau_{ij}^k$ denotes the pheromone of the ant_k left on the edge $edge_{ij}$ between $city_i$ and $city_j$ in this iteration, which is given by:

$$\Delta\tau_{ij}^k = \begin{cases} \frac{Q}{L_k}, & if\ ant_k\ passed\ edge_{ij} \\ 0, & other \end{cases} \tag{12}$$

where Q is a hyperparameter, L_k is the path length of the ant_k in this iteration.

B. Graph Convolutional Network based Ant Colony Optimization

Fig. 1 presents the framework of the proposed GCN-ACO algorithm which comprises two stages. Firstly, we train a

GCN model to predict the probability of each edge in the graph of TSP and VRP problem instances belonging to the optimal route. Subsequently, the predicted probability values are leveraged to enhance the performance of ACO in finding high-quality solutions.

In the first stage of the proposed GCN-ACO algorithm, we utilize the GCN model outlined in [24]. Each model comprises 30 graph convolutional layers and 3 layers in the MLP with hidden dimension h = 300 for each layer. Separate training, validation, and testing datasets are created for TSP problems with 20, 50, and 100 nodes, as well as VRP problems with 100 nodes. For TSP instances, n nodes are randomly sampled within the unit square and $node_i$ is represented by coordinates (x_i, y_i). The solutions produced by the exact solver Concorde[1] are used as the ground-truth solutions. For the VRP instances, appropriate modifications are made to the model presented in [24]. In addition to randomly sampling n nodes within the unit square, a special depot node dep is included [25]. Each node $node_i$ has an extra feature d_i that represents the required capacity for $node_i$, in addition to its coordinates, where $d_i \leq C$ and $d_{dep} = 0$. For the VRP model, training labels from example solutions generated by LKH [7] are used, although these solutions are sub-optimal, they still provide valuable information during the training process. Within the sample labels, edges present in the optimal path are marked as "1", while absent edges are labeled as "0". Consequently, GCN models can be trained using these labels. The resulting model is capable of predicting the probability heatmap h, which represents the probability (ranging from 0 to 1) that each edge belongs to the optimal route. Fig. 1 (b) demonstrates that the darkness of each edge's color indicates a higher probability of belonging to the optimal path.

The pheromone matrix τ is a crucial component of ACO, upon which the search quality of ACO heavily hinges. In traditional ACO, τ values are typically initialized uniformly and evolve with the iterations of the ACO. Therefore, in the second stage of our approach, we incorporate the probability values h predicted by the GCN model into the ACO search process to enhance its performance. Firstly, we consider initializing the τ values in the AS directly using the predicted probability heatmap h from the model, i.e., $\tau_0 = h$. This approach can provide good guidance for the ant colony during its initial stage, preventing blind searching and hastening its convergence rate. However, in the above approach, the solution quality of ACO heavily hinges on the accuracy of the network model prediction, which can sometimes mislead it to converge to local optima and only be effective at the initial stage. As a result, we explore a hybrid approach that combines the predicted probability h and pheromone τ, as shown in formula (13). In addition, we also explore the method of combining GCN with other variants of ant colony optimization such as MMAS, referred to as GCN-MMAS, which restricts the pheromone values τ and predicted probabilities h within the range of $[\tau_{min}, \tau_{max}]$.

$$p_{ij}^k(t) = \begin{cases} \frac{[\tau_{ij}(t)]^\alpha \cdot [h_{ij}(t)]^\gamma \cdot [\eta_{ij}(t)]^\beta}{\sum_{s \in J_k} [\tau_{is}(t)]^\alpha \cdot [h_{is}(t)]^\gamma \cdot [\eta_{is}(t)]^\beta}, & j \in J_k \\ 0, & other \end{cases} \tag{13}$$

V. Experiments

A. Experimental Environment

All algorithms were implemented using python 3.9 in pytorch. During the first stage, the GCN model was trained on a server equipped with a GeForce RTX 3090 GPU, and inference and testing were conducted on a PC desktop equipped with a GeForce RTX 3060 GPU, Intel Core i5-11500 @ 2.70GHz CPU, and 32GB memory.

B. Experimental Parameters

For better comparison of algorithm performance, we set the basic parameters in all ACO and its variants involved in the experiment to be the same. The number of ants m is 20, the number of ant colony iterations n is 20, α is 1, β is 3, γ is 1, the pheromone evaporation factor ρ is 0.3, and the hyperparameter Q is 0.1. The initial pheromone τ_0 is set to 1 in traditional ACO, while in MMAS, τ_{min} is 0.01 and τ_{max} is 1.

C. Experimental Setup and Results

(1) Effectiveness of GCN-ACO

To validate the effectiveness of the proposed algorithm, it is tested on TSP problems with node numbers of 20, 50 and 100, as well as on VRP problems with a node number of 100, which are represented as TSP20, TSP50, TSP100 and VRP100 respectively. To assess the predictive accuracy of the GCN model, we adopted a greedy strategy based on the GCN predictions to generate an optimal solution. At each step, it selected the edge with the highest probability value, enabling to construct a complete solution named as GCN-Greedy. Additionally, we investigated three approaches to integrate GCN with ACO, which are shown as follows:

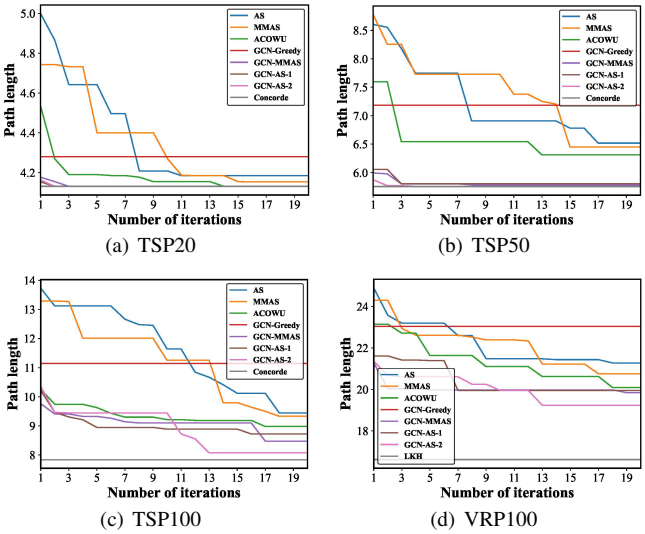
- GCN-AS-1: using the probability values predicted by GCN as the initial pheromone of AS;
- GCN-AS-2: incorporating the probability values predicted by GCN into the probability transition formula of AS;
- GCN-MMAS: combining the probability values predicted by GCN with MMAS.

During the testing process, 100 test instances were randomly generated for each problem, and all algorithms were independently run 10 times on each test case. Concorde's solutions were selected as the optimal values (L_{opt}) for the TSP problems (with LKH's solutions used as the optimal values for VRP problems). Our proposed algorithms were compared with AS, MMAS, and ACOWU [20] in terms of the best path length ($Best$), the average path length (Avg) and the standard deviation (Std) across ten trials, the gap (Gap) between the average path length and the optimal value (as shown in Equation (14), where L_m is the average path length

[1]https://www.math.uwaterloo.ca/tsp/concorde/

TABLE I
RESULTS OF DIFFERENT METHODS IN SOLVING TSP20, TSP50, TSP100, AND VRP100 TEST INSTANCES.

Problem	Metric	Opt	AS	MMAS	ACOWU	GCN-Greedy	GCN-MMAS	GCN-AS-1	GCN-AS-2
TSP20	Best	4.1313	4.1537	4.1313	4.1313	4.1313	4.1313	4.1313	4.1313
	Avg	4.1313	4.1746	4.1592	**4.1313**	4.2807	**4.1313**	**4.1313**	**4.1313**
	Std	0	0.0395	0.0167	0	0.3148	0	0	0
	Gap	0.00%	1.05%	0.67%	**0.00%**	3.62%	**0.00%**	**0.00%**	**0.00%**
	Iteration	-	12.7	14.2	12.6	-	2.7	1.8	3.2
	Time(s)	0.036	0.1748	0.1898	0.1776	0.0227	0.0361	0.0248	0.0444
TSP50	Best	5.7543	6.2759	6.2319	6.1888	6.2882	5.7591	5.7566	5.7543
	Avg	5.7543	6.4533	6.4127	6.3192	7.185	5.8017	5.8155	**5.7561**
	Std	0	0.1693	0.0864	0.0904	0.6362	0.0349	0.0554	0.0009
	Gap	0.00%	12.14%	11.44%	9.82%	24.86%	0.82%	1.06%	**0.03%**
	Iteration	-	17.4	18.1	16.1	-	7.0	4.0	5.1
	Time(s)	0.08	1.3419	1.4026	1.4265	0.0293	0.5896	0.3101	0.4884
TSP100	Best	7.8393	9.2199	9.1668	8.3292	10.4463	8.1801	8.4101	8.0828
	Avg	7.8393	9.6063	9.4577	8.9967	11.143	8.6342	8.7511	**8.3136**
	Std	0	0.2037	0.2073	0.2725	0.6003	0.2293	0.1848	0.1584
	Gap	0.00%	22.53%	20.64%	14.76%	42.14%	10.13%	11.63%	**6.04%**
	Iteration	-	18.6	19.1	18.3	-	12.7	13.4	13.6
	Time(s)	0.36	6.9343	6.609	6.1878	0.0427	4.2525	4.657	5.0124
VRP100	Best	16.6204	20.6786	19.4028	19.8536	21.1906	19.1627	19.6601	18.445
	Avg	16.6204	21.1713	20.5195	20.227	23.0409	19.767	19.9274	**19.113**
	Std	0	0.3451	0.4009	0.2459	1.0361	0.3027	0.2242	0.2741
	Gap	0.00%	27.38%	23.45%	21.69%	38.62%	18.93%	19.89%	**14.99%**
	Iteration	-	17.1	16.1	15.8	-	13.4	11.4	9.8
	Time(s)	4.6874	5.5735	5.2781	5.0479	0.0479	3.9887	3.3324	2.9567



(a) TSP20 (b) TSP50

(c) TSP100 (d) VRP100

Fig. 2. The convergence curves of different methods on TSP20, TSP50, TSP100, and VRP100 test instances.

of other methods), convergence iteration times $(Iteration)$, and computation time $(Time)$. Here, the models used in GCN are trained on datasets corresponding to the size of the problems.

$$gap = \frac{L_m - L_{opt}}{L_{opt}} \times 100\% \tag{14}$$

1) Comparisons with the traditional methods: Table I presents the quantitative results of various metrics for solving TSP20, TSP50, TSP100 and VRP100 test instances with different methods. Based on the table, it is evident that the AS and MMAS methods still have a slight gap with the optimal solution for the TSP20 problem, while the GCN-ACO methods (GCN-AS-1, GCN-AS-2, and GCN-MMAS) always find the optimal solution. For larger TSP instances such as TSP50 and TSP100, the gaps between the solutions found by different methods and the optimal solution tend to increase as the problem size grows. However, GCN-ACO still significantly outperforms traditional ACO and its variants. Due to the complexity of the VRP100 problem, AS, MMAS, and ACOWU all have gaps of over 20% compared to the optimal solution, while GCN-ACO has gaps below 20%. In particular, the gap for GCN-AS-2 is as low as 14.99%, which is 12.39% less than AS. Fig. 2 presents the convergence curves on a test case of the four problems for these methods. It can be observed that our proposed method is capable of finding a better solution from the beginning and converging to a better solution with fewer iterations.

2) Comparisons between different integration methods: The differences between the three integration methods proposed can be observed from the last three columns in Table I. GCN-MMAS achieves better results than GCN-AS-1 because it constrains both the predicted probability values and pheromone values within a certain range, avoiding the situation

TABLE II
RESULTS OF AS AND GCN-AS-2 BASED ON VRP100 MODEL IN SOLVING VRP INSTANCES.

Instance	Opt	AS			GCN-AS-2		
		Best	Avg±Std	Gap	Best	Avg±Std	Gap
A-n32-k5.vrp	784	838.63	885.69±15.46	12.97%	834.95	836.85±1.58	**6.74%**
A-n45-k7.vrp	1146	1289.18	1307.11±13.19	14.05%	1223.7	1246.97±6.47	**8.81%**
A-n60-k9.vrp	1354	1524.35	1545.32±15.11	14.13%	1465.64	1486.01±4.81	**9.75%**
A-n80-k10.vrp	1763	2041.24	2057.04±11.64	16.67%	1923.77	1970.85±11.01	**11.79%**
E-n101-k8.vrp	815	1010.89	1040.56±20.98	27.67%	912.13	934.23±13.35	**14.63%**
M-n151-k12.vrp	1015	1347.51	1362.63±10.96	34.24%	1188.58	1202.87±15.03	**18.51%**

TABLE III
RESULTS OF GCN-AS-2 BASED ON DIFFERENT MODELS IN SOLVING TSP INSTANCES.

Instance	Opt	TSP20 Model			TSP50 Model			TSP100 Model		
		Best	Avg±Std	Gap	Best	Avg±Std	Gap	Best	Avg±Std	Gap
Bays29.tsp	2020	2020	2030.51±0.0053	**0.52%**	2020	2037.68±0.0413	0.87%	2037.42	2051.4±0.0499	1.55%
Att48.tsp	10628	10879.24	11052.64±0.0652	3.99%	10670.07	10810.45±0.0579	**1.70%**	10791.78	10973.79±0.0651	3.25%
berlin52.tsp	7542	7644.73	7824.08±0.0804	3.74%	7542	7598.96±0.0387	**0.75%**	7542	7687.63±0.0804	1.93%
St70.tsp	675	737.97	759.67±0.1491	12.54%	699.75	732.05±0.1515	8.45%	700.33	719.19±0.1519	**6.54%**
kroA100.tsp	21282	21578.12	22785.23±0.2536	7.06%	22009.42	22523.62±0.1195	5.83%	21282	22000.66±0.1908	**3.37%**
Ch130.tsp	6110	6710.47	6904.34±0.2027	13.00%	6585.19	6836.63±0.1859	11.89%	6541.45	6694.29±0.1131	**9.56%**

where there is too much or almost no pheromone on certain edges. According to the results, GCN-AS-2 outperforms the other two methods. The reason may be that it does not solely rely on the GCN predicted probabilities as the initial pheromone values, but instead, it combines them with the pheromone. This approach is particularly beneficial for large-scale problems, where the GCN predicted probabilities may not be accurate enough, potentially leading the ants astray. By incorporating the predicted probabilities of GCN with the pheromone, the combined method reduces the error and is more robust.

3) Ablation analysis: To further demonstrate the significance of probability heatmaps in ant colony optimization, we conducted an ablation analysis of our proposed method. GCN-AS-2 combines probability heatmaps and pheromone of AS. Table I shows that using only probability heatmaps and employing a greedy approach to search for solutions results in considerably worse solutions than the optimal ones in a very short time. On the other hand, AS without heatmaps yields solutions that are relatively closer to the optimal ones, but it requires a much longer time to search. Therefore, GCN-AS-2 combines the strengths of both approaches, enabling it to find better solutions in a shorter time.

(2) Generalization of GCN models

The ability of an algorithm to generalize over problems of varying sizes is a crucial performance indicator. Since the GCN model parameters are independent of problem size, we can theoretically apply the trained model to problems of any size. Consequently, we evaluated the generalization ability of the model by conducting the following experiments.

1) Generalization to problems of various scales: To evaluate the model's generalization ability on problems of varying sizes, we tested GCN-AS-2 which utilizes a GCN model trained on VRP100 dataset on several VRPLIB benchmarks[2], where n and k represent the number of nodes and vehicles respectively. Based on the experimental results presented in Table II, it was observed that GCN-AS-2 performed better than AS on problems of different sizes, with the difference between their performances increasing as problem size increased. Furthermore, experiments have indicated that models trained on small-scale datasets can be useful in solving large-scale problems. For instance, GCN-AS-2, employing VRP100 model, achieved a 15.73% reduction compared to AS for M-n151-k12.vrp instance with 151 nodes.

2) Generalization of different models: Additionally, we conducted tests on standard TSPLIB instances[3] to explore the relationship between different models and problem sizes using models trained on various TSP datasets. The digits in the instance names represent the number of nodes. As presented in Table III, GCN-AS-2 with models trained on TSP20, TSP50 and TSP100 datasets (labeled as TSP20 Model, TSP50 Model and TSP100 Model respectively) were tested on TSP instances ranging from 29 to 130, with each test case being executed 10 times. The experimental results demonstrate that the models trained using different datasets were effective in solving these instances. Specifically, the best results were obtained when the size of the models was close to the size of the problems.

[2]http://vrp.atd-lab.inf.puc-rio.br/index.php/en/

[3]http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/tsp/

VI. CONCLUSIONS

In this paper, we propose a hybrid ant colony optimization based on graph convolutional networks which guides ant colony search using predicted heatmaps to address the robot task allocation problems, which are formulated as the TSP and the VRP. Experimental results demonstrate that the proposed method significantly outperforms traditional ACO in terms of its convergence speed and solution quality. Moreover, we investigate three different combinations and test the generalization capability of the proposed approach on problems of different scales. Future work will further explore more effective ways to combine GCN and ACO to apply our proposed method to larger-scale problems.

REFERENCES

[1] Y. Shi, B. Hu, and R. Huang, "Task allocation and path planning of many robots with motion uncertainty in a warehouse environment," in *2021 IEEE International Conference on Real-time Computing and Robotics (RCAR)*. IEEE, 2021, pp. 776–781.

[2] C. Wei, Z. Ji, and B. Cai, "Particle swarm optimization for cooperative multi-robot task allocation: a multi-objective approach," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2530–2537, 2020.

[3] L. Huang, Y. Ding, M. Zhou, Y. Jin, and K. Hao, "Multiple-solution optimization strategy for multirobot task allocation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 11, pp. 4283–4294, 2018.

[4] S. Ma, W. Guo, R. Song, and Y. Liu, "Unsupervised learning based coordinated multi-task allocation for unmanned surface vehicles," *Neurocomputing*, vol. 420, pp. 227–245, 2021.

[5] Y. Liu, R. Song, R. Bucknall, and X. Zhang, "Intelligent multi-task allocation and planning for multiple unmanned surface vehicles (usvs) using self-organising maps and fast marching method," *Information Sciences*, vol. 496, pp. 180–197, 2019.

[6] R. Kuo, S.-H. Lu, P.-Y. Lai, and S. T. W. Mara, "Vehicle routing problem with drones considering time windows," *Expert Systems with Applications*, vol. 191, p. 116264, 2022.

[7] J. Zheng, K. He, J. Zhou, Y. Jin, and C.-M. Li, "Reinforced lin–kernighan–helsgaun algorithms for the traveling salesman problems," *Knowledge-Based Systems*, vol. 260, p. 110144, 2023.

[8] M. Karimi-Mamaghan, M. Mohammadi, P. Meyer, A. M. Karimi-Mamaghan, and E.-G. Talbi, "Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: A state-of-the-art," *European Journal of Operational Research*, vol. 296, no. 2, pp. 393–422, 2022.

[9] Y. Shen, Y. Sun, X. Li, A. Eberhard, and A. Ernst, "Adaptive solution prediction for combinatorial optimization," *European Journal of Operational Research*, vol. 309, no. 3, pp. 1392–1408, 2023.

[10] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE computational intelligence magazine*, vol. 1, no. 4, pp. 28–39, 2006.

[11] X. Li and Z. Liu, "Research on improvement of ant colony algorithm for multi-robot task allocation," in *2019 IEEE 8th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*. IEEE, 2019, pp. 1315–1319.

[12] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[13] X.-F. Liu, B.-C. Lin, Z.-H. Zhan, S.-W. Jeon, and J. Zhang, "An efficient ant colony system for multi-robot task allocation with large-scale cooperative tasks and precedence constraints," in *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2021, pp. 1–8.

[14] M. Gasse, D. Chételat, N. Ferroni, L. Charlin, and A. Lodi, "Exact combinatorial optimization with graph convolutional neural networks," *Advances in neural information processing systems*, vol. 32, 2019.

[15] S. Gao, J. Wu, and J. Ai, "Multi-uav reconnaissance task allocation for heterogeneous targets using grouping ant colony optimization algorithm," *Soft Computing*, vol. 25, pp. 7155–7167, 2021.

[16] N. Geng, Z. Chen, Q. A. Nguyen, and D. Gong, "Particle swarm optimization algorithm for the optimization of rescue task allocation with uncertain time constraints," *Complex & Intelligent Systems*, vol. 7, pp. 873–890, 2021.

[17] R. Patel, E. Rudnick-Cohen, S. Azarm, M. Otte, H. Xu, and J. W. Herrmann, "Decentralized task allocation in multi-agent systems using a decentralized genetic algorithm," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 3770–3776.

[18] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 1, pp. 29–41, 1996.

[19] T. Stützle and H. H. Hoos, "Max–min ant system," *Future generation computer systems*, vol. 16, no. 8, pp. 889–914, 2000.

[20] M. Neroni, "Ant colony optimization with warm-up," *Algorithms*, vol. 14, no. 10, p. 295, 2021.

[21] G. K. Baydogmus, "Solution for tsp/mtsp with an improved parallel clustering and elitist aco," *Computer Science and Information Systems*, vol. 20, no. 1, pp. 195–214, 2023.

[22] Y. Sun, S. Wang, Y. Shen, X. Li, A. T. Ernst, and M. Kirley, "Boosting ant colony optimization via solution prediction and machine learning," *Computers & Operations Research*, vol. 143, p. 105769, 2022.

[23] B. Li, G. Wu, Y. He, M. Fan, and W. Pedrycz, "An overview and experimental study of learning-based optimization algorithms for the vehicle routing problem," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, no. 7, pp. 1115–1138, 2022.

[24] C. K. Joshi, T. Laurent, and X. Bresson, "An efficient graph convolutional network technique for the travelling salesman problem," *arXiv preprint arXiv:1906.01227*, 2019.

[25] W. Kool, H. van Hoof, J. Gromicho, and M. Welling, "Deep policy dynamic programming for vehicle routing problems," in *Integration of Constraint Programming, Artificial Intelligence, and Operations Research: 19th International Conference, CPAIOR 2022*. Springer, 2022, pp. 190–213.