Balancing Task Allocation in Multi-robot Systems Using adpK-Means Clustering Algorithm

Ling Chong, Qingjie Zhao(✉), and Kairen Fang

Beijing Laboratory of Intelligent Information Technology, School of Computer Science, Beijing Institute of Technology, Beijing 100081, China
{chongling,zhaoqj,3220180793}@bit.edu.cn

Abstract. Multi-robot systems are becoming more and more significant in industrial, where allocating tasks for every robot in a reasonable way is a tedious process. The current research mainly focused on reducing the distance between robots and tasks while ignoring the balance of workloads between robots. To address the aforementioned issues, this paper proposes an adaptive K-means clustering algorithm (adpK-means) in order to control a team of robots to accomplish all tasks with a good balance and at a minimal cost. Compared with the K-means clustering algorithm, our proposed algorithm has better performance, where through adaptive dynamic scaling of the clustering space in the iterative process, multiple robots can complete missions with well-distributed workloads. The experimental results show that the algorithm effectively reduces the total energy consumption of the entire robot system and ensures that the tasks of robots are comparative.

Keywords: Multi-robot · Task allocation · adpK-means clustering algorithm

1 Introduction

With the development of modern industry and manufacturing, industrial robots are widely used in automobiles, electronics, logistics, metals, food, and chemicals [1]. In recent years, with the highly coupled and coordinated development of multiple disciplines such as computer technology, electronic communication, automation control, artificial intelligence, chips, robots have almost replaced highly repetitive and dangerous tasks [2]. However, the single-robot system cannot undertake the complex task requirements of industry and manufacturing because of its weak individual perception and processing information ability and single task execution ability, and the multi-robot system has emerged as the times require [3]. The multi-robot system is composed of multiple intelligent robots that can perceive the environment and perform different tasks. The collaborative work of multi-robots can make full use of the advantages of multi-machine collaboration and carry out industrial production more efficiently and stably,

This work was supported by the National Key R&D Program of China (No. 2017YFB1303300 and 2019YFA0706500).

F. Sun et al. (Eds.): ICCSIP 2020, CCIS 1397, pp. 227–242, 2021.
https://doi.org/10.1007/978-981-16-2336-3_21

but it also brings scholars [4]. Here comes a huge challenge, that is, how to allocate tasks for the multi-robot system.

The core problem of the multi-robot system's task allocation is to make the task allocation of different robots more balanced as much as possible to minimize the overall consumption of the multi-robot system. The collaborative work of multi-robot systems usually has two stages [5]. The first link is multi-robot task allocation, and the second link is multi-robot path planning. Usually, the task allocation result of the first stage directly affects whether the second stage can be solved to obtain the lowest cost work path. If the distribution is unreasonable or unbalanced, it isn't easy to find an efficient path planning sequence in the second stage of planning [6]. Multiple Traveling Salesman Problem [7] (Multiple Traveling Salesman Problem, MTSP) is a way to model this problem. This modelling method compares each robot to a travelling salesman, and the waypoint represents the city location that the traveling salesman needs to visit. The goal of optimization is to meet the constraint conditions; all path points will be called, and optimize the objective function [8].

Many types of research on task allocation for multi-robots system have been performed. For instance, these studies include combinatorial auction algorithm [9], genetic algorithm [10], agent-based algorithm [11], pattern formation algorithm [12], the graph matching algorithm [13] and so on. For example, Xu et al. [14] proposed a two-stage MTSP solution method, which clustered all cities through the K-means clustering method. The number of clusters is the number of traveling salesmen. During the clustering, the workload of a different travelling salesman is guaranteed to be as balanced as possible. The upper limit is set for the number of cities allocated by each travelling salesman after clustering to make the distribution result more reasonable. In the second stage, a genetic algorithm is used to plan and solve the results after clustering. Xie [15] uses the multi-knapsack algorithm to allocate the welding path points and improves the genetic algorithm to realize the path planning of a single robot. Shi Dechao [16] also used the multi-knapsack algorithm to learn the distribution of welding points and added the limit of the number of welding points so that the workload of each robot could be balanced, which solved the problem of welding point path distribution in white body welding. In the planning stage, detailed parameter analysis and adjustment are carried out according to the actual scene to realize welding path planning. Burger et al. [17] proposed a dichotomous variable method based on 2-index. Each time the next path point is selected, the selection is based on the current path point, which reduces the complexity of running time and improves the efficiency of the algorithm. Kitjacharoenchai et al. [18] used K-means clustering pre-allocation for the logistics and distribution of trucks to obtain the set of goods that each car needs to be responsible for. In planning, the idea of a greedy strategy is adopted, and the cargo location with the shortest distance from the current location is selected as the next route point.

However, the current attempts made by the researchers concentrate only on minimizing the distance between the robots and the tasks, and not much importance is given to the utilization of the robots. In industrial production, multi-robot systems mostly work in an assembly-line manner. Uneven distribution of tasks among multiple robots will result in higher system consumption. The K-means algorithm can divide all vectors in the clustering space into K clusters. Formally, because of this, the clustering algorithm

can be used to solve the multi-robot task allocation problem. However, it has limitations. The algorithm is relatively affected by initialization and is more susceptible to outliers. Therefore, it cannot guarantee the balance of multi-robot task allocation. To overcome these deficiencies, this paper proposes an adpK-means clustering algorithm through adaptive dynamic scaling of the clustering space in the iterative process.

2 Problem Statement

The multi-traveling salesman problem means that multiple traveling salesmen traverse all cities under the condition of satisfying the constraints, and solve the planning path of each traveling salesman when the objective function is optimized [19]. The path planning problem of multi-robots is that n robots are responsible for executing m task points. After completing the task, each robot returns to its starting position to ensure that each task point will be executed by the robot and cannot be performed repeatedly. In this article, we define it as a multi-source closed-loop multi-traveling salesman problem. The traveling salesman can be abstracted as a robot, and the task point is abstracted as the city that the traveling salesman wants to visit. The following will conduct mathematical modeling and research on this problem.

There are two commonly used evaluation methods in the multi-source closed-loop multi-traveling salesman problem: minimizing the maximum cost *maxspan* and minimizing the total cost *mincost*. Suppose there are a total of n traveling salesmen, and the cost of the i-th traveling salesman is $cost_i$ ◦The description of *maxspan* is shown in formula (1), and the description of *mincost* is shown in formula (2).

$$maxspan = max\{cost_1, cost_2, \cdots, cost_n\} \tag{1}$$

$$mincost = \sum\nolimits_{i=1}^{n} cost_i \tag{2}$$

Because the engineering problem faced by this article is a multi-robot collaborative operation problem, in the actual operation process, only when all the robots complete the operation can the operation of the next workpiece be successfully carried out. Considering that multi-robot systems are commonly oriented to industries, the robot's working time is based on the robot with the longest working time, so we use the minimized maximum cost *maxspan* as the objective function.

The specific mathematical definition of the multi-source closed-loop multi-traveling salesman problem is given here: Given a set of n cities, the number of traveling salesmen is t, and the distance between city i and city j is defined as $d(i, j)$. The number of cities allocated by the k-th traveling salesman is $count_k$, and the sequence of cities visited is $Tout_k = \left(c_1^k, c_2^k, \cdots, c_{count_k}^k\right)$, where $\sum_{k=1}^{t} count_k = n$, each city will be Traverse and only traverse once, according to formula (3), calculate the path length $cost_k$ of the k-th traveling salesman as shown in formula (3), and the objective function calculation is shown in formula (2).

$$cost_k = \sum\nolimits_{i=j}^{count_k - 1} d\left(c_j^k, c_{j+1}^k\right) + d\left(c_1^k, c_{count_k}^k\right) \tag{3}$$

Task allocation is to allocate tasks to the corresponding traveling salesman for planning, simplifying complex problems, and turning them into general traveling salesman problems, which can significantly reduce the complexity of the problem and improve calculation efficiency. The path point allocation process will have different restrictions according to actual application scenarios. The scenario in this article is the collaborative work of multiple robots, and it is stipulated that each robot has the same structure and the same type of task point. Therefore, any task point can be assigned to any robot, and there is no upper limit on the number of task points for each robot. In this paper, our objective function metric is to minimize the maximum cost *maxspan*, which is also out of consideration for practical problems. The multi-robot execution operation process is an assembly line operation process in an industrial environment. Therefore, optimizing the robot with the longest working time and reducing its running time can improve the efficiency of industrial production.

3 AdpK-Means Clustering Algorithm

3.1 K-Means Clustering Algorithm

Researchers related to task allocation algorithm used K-means clustering or other clustering algorithms [20] to solve the problem, and achieved good results. The clustering algorithm is an unsupervised algorithm used for sample classification problems in machine learning [21]. K-means clustering algorithm (K-means) is the most common, most commonly used, and most effective clustering algorithm. The K-means clustering result divides all vectors in the clustering space into K clusters [22]. It is precisely because of this characteristic that the clustering algorithm can solve the path point allocation problem in multi-robot path planning. Suppose the number of robots is K, then the goal of clustering is to divide the path points into K sets, and one robot is responsible for planning the path points in each group. The specific calculation process of K-means is described in detail in Algorithm 1.

Algorithm 1: K-means clustering algorithm

1 Set the number of cluster center points K, select K points as the initial points
2 **while** The category of all points remains unchanged **do**
3 Calculate the Euclidean distance from all points to K cluster center points
4 Return points: the attribution category of each store is the closest cluster center point
5 Recalculate the new cluster center point for each cluster set: take the average
6 **end while**

Euclidean distance is commonly used in K-means clustering algorithm to estimate the similarity between different samples [23]. Also, there are Manhattan distance [24], cosine distance [25], etc. The distance measurement methods used for different types of data and tasks are other. In this article, we plan the optimal path of the robot in the geometric space. The length of the way of the robot between different path points is calculated using Euclidean distance. Therefore, in this article, we also use Euclidean distance to estimate the similarity between different samples.

The K-means algorithm also has some limitations. The algorithm is more affected by initialization, different initialization will bring different results, and it is more susceptible to outliers [26], Although the subsequent improved K-means++ algorithm [27] improves this deficiency by selecting the initial point based on probability, the influence of initialization still exists. Moreover, the distribution result obtained by K-means cannot guarantee the balance of tasks between robots. During the operation of the algorithm, which set the waypoint belongs to is determined by the distance from the waypoint to the central point of the set, which cannot directly reflect different groups. Whether the workload is balanced, especially when there is a long tail effect of outliers in the data distribution, the imbalance of distribution is more serious. On this point, related experiments have also been carried out in this article, which proves that the distribution results obtained by the K-means clustering algorithm have room for improvement in the distribution balance. However, it has particular applicability as a waypoint allocation scheme in the multiple traveling salesman problem. It is necessary to avoid collisions as much as possible when multi-robots work together. The different path point sets resulting from K-means clustering are relatively independent, and there is no overlap. This can make the planned paths of other robots fewer collision problems, which is very beneficial for the subsequent collision detection optimization steps. Therefore, this paper proposes an adpK-means clustering algorithm, which achieves the balanced distribution of path points by adaptively scaling the clustering space in the iterative process, and proves the effectiveness of the algorithm through experiments.

3.2 AdpK-Means Clustering Algorithm Task Allocation

The adpK-means algorithm is an iterative optimization algorithm, and the running process is shown in Fig. 1. The adpK-means algorithm will count the current clustering results during each iteration and dynamically scale the clustering space according to the products until the termination condition is met. The optimal outcome is output. This way of dynamic scaling and iterative optimization can make the distribution of path points more balanced.

In order to describe the operation process of the algorithm more clearly, this article will define the implementation steps and details of the adpK-means clustering algorithm in a two-dimensional space. The algorithm operation flow in the three-dimensional space is the same as that in the two-dimensional space, and only one dimension parameter needs to be added. The specific operation flow is described in detail in Algorithm 1. The number of clustering categories K is the number of robots, and the workload of the robots is estimated as the length of the path sequence estimated by the greedy strategy. From the description of the algorithm, it can be seen that in each iteration, the algorithm shrinks the path points in the set with the smallest workload and enlarges the points in the group with the most massive workload. After shrinking and zooming in, the distribution of path points in the clustering space changes, and the results will also change after K-means clustering. More path points will be allocated in the set with the smallest workload, and the number of path points in the set with the largest workload will be reduced. This is how the adaptive zoom operation makes the workload of different robots more balanced.

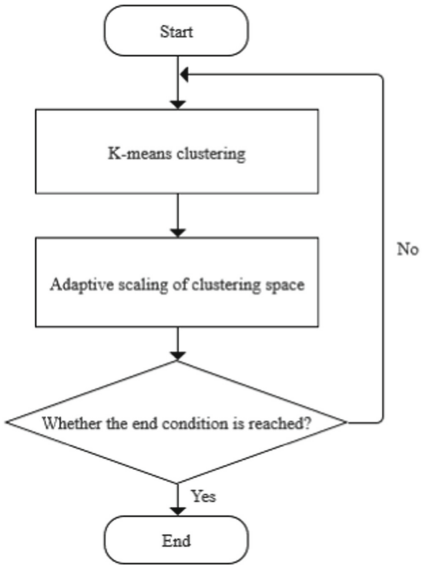


Fig. 1. Flow chart of adpK-means clustering algorithm.

Algorithm 2: AdpK-means clustering algorithm

1 Set the maximum number of iterations $Kmeans\_\max\_iter$, adaptive scaling $Scale$, the number of categories is K

2 **while** Maximum number of iterations not reached **do**

3 Use K-means algorithm to cluster all path points into Ksets

4 Calculate the planned path obtained according to the greedy strategy in each set, and record the path length as l_i, where $i = \{1,2,\cdots,K\}$

5 In the set l_i, the set corresponding to the maximum value is the set with the largest workload. Calculate the center point of all path points in the set as $P_{center} = (X_c - Y_c)$,and perform the zoom operation on the path points in the set:

$$P_x^i = \left(P_x^i - X_c\right) \times Scale + X_c$$
$$P_y^i = \left(P_y^i - Y_c\right) \times Scale + Y_c$$

6 In the set l_i, the set corresponding to the minimum value is the set with the smallest workload. Calculate the center point of all path points in the set as $P_{center} = (X_c - Y_c)$, and perform the reduction operation on the path points in the set:

$$P_x^i = \left(P_x^i - X_c\right) \div Scale + X_c$$
$$P_y^i = \left(P_y^i - Y_c\right) \div Scale + Y_c$$

7 **end while**

8 Output the final classification result

To better illustrate the effect of adaptive scaling on the change of clustering space and the balanced distribution of path points, this paper visualizes the scaling adjustment process of the algorithm in the experiment of the TSPLIB dataset [28] pr76.tsp in Fig. 2. In this data set, we set the number of robots to 4. After a K-means clustering, four point

sets are obtained, marked with different colors in the figure. As described in Algorithm 2, a greedy strategy is used to find a path for other sets, and the workload of the robots in the set is estimated according to the length of the path. The light red set is the set with the largest amount of work path, and the light black set is the set with the smallest amount of work. Algorithm 2 describes a scaling process. In the example, the center point is calculated for the light red point group, and all points in the set are spread out with the scaling factor of *Scale*. In Fig. 2. the spread point set is marked in dark red. The light black point set also calculates the position of the center point, and all points in the set are contracted inward by the scaling factor of *Scale*. The dark black point set in the figure is the result of the contraction.

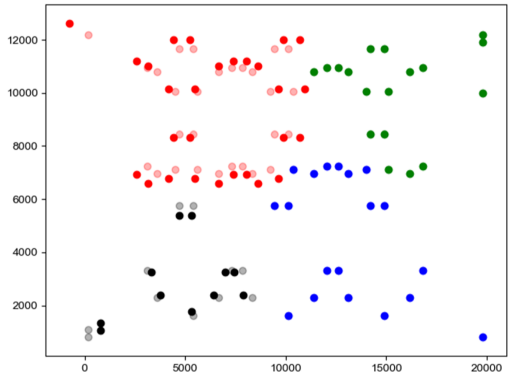


Fig. 2. Schematic diagram of the change process of task points in adpK-means clustering.

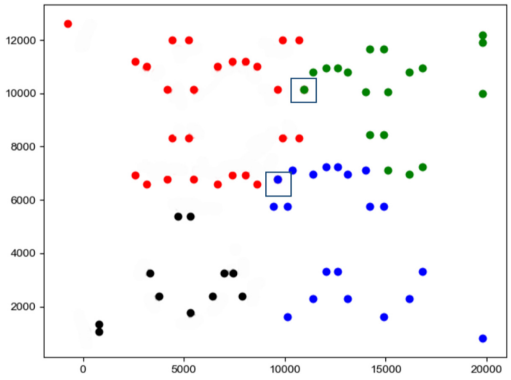


Fig. 3. Clustering result graph after change.

The points in Fig. 3 are the positions of all path points after dynamic shrinkage. The K-means algorithm is used to cluster again on the new data distribution, and you can see that the clustering results have changed. One of the path points in the red set is allocated

to the green set, and one is assigned to the blue group. In Fig. 3, the waypoints where the attribution category changes are marked with boxes. The red waypoint set is the set with the most massive workload estimated in the previous iteration. After one adjustment, the samples in the red waypoint set are allocated to other robots, and the workload of the robots in this set will be reduced. The robot workload will increase. From the example, we can intuitively see the change of adaptive clustering to the clustering results. Among them, the scaling factor *Scale* plays a vital role. Adaptive scaling does not guarantee that you can get better than the last time. As a result, setting an appropriate *Scale* value can make the algorithm search for clustering results with a balanced workload faster.

3.3 Path Planning

To verify that the task allocation algorithm proposed in this paper can reduce the overall consumption of the system after the task allocation is completed, the same path planning algorithm is used, namely the discrete multi-group fruit fly algorithm (GA-DMFOA) fused with genetic operators, Carry out path planning. Path planning is to transform the problem into a general traveling salesman problem to solve it according to the distribution result. The GA-DMFOA algorithm [29, 30] is used to complete the path planning of each robot. As shown in Fig. 4, taking ten path points {1,2,3,4,5,6,7,8,9,10} as an example, and three robots as an example, the distribution result is {1,2,3},{4,5,6,7},{8,9,10} three sets, each robot is responsible for planning a group, and the intended path of each robot is obtained through the GA-DMFOA algorithm. To verify that the task allocation algorithm proposed in this paper can reduce the overall consumption of the system after the task allocation is completed, the same path planning algorithm is used, namely the discrete multi-group fruit fly algorithm (GA-DMFOA) fused with genetic operators, Carry out path planning.

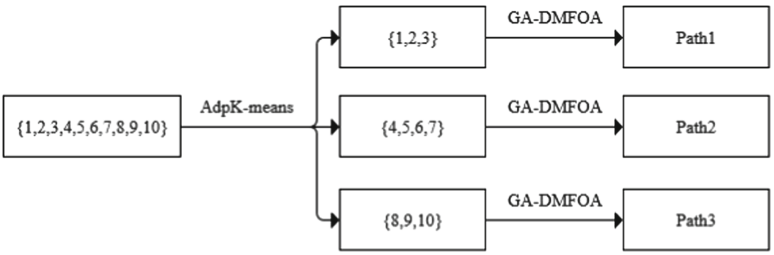


Fig. 4. Schematic diagram of path planning example.

3.4 Algorithm Overall Steps

This paper introduces the relevant steps and implementation details of the adpK-means clustering algorithm and completes the balanced allocation of path points. The multi-traveling salesman problem is decomposed into a single traveling salesman problem in a two-stage manner. The improved fruit fly optimization algorithm GA-DMFOA is used

to plan the path of each robot, and an effective solution to the multi-robot path planning problem is proposed. The parameters used in the multi-robot path planning algorithm are shown in Table 1, where Algorithm 3 describes the solution process of the multi-robot path planning problem. Figures 5(a) and 5(b) show the comparison of the results before and after optimization using the delayed wait strategy when the distance between robots is less than the safe distance l_safe.

Table 1. Multi-robot path planning parameter description table.

Parameter	Description
*Kmeans*_max_*iter*	Maximum iteration times of adpK-means clustering
Scale	Adaptive scaling
K	Number of robots
l_{safe}	Safe distance between multiple robots
v	The speed of the robot
M	Number of individuals in the population
MaxIter	The maximum number of iterations
Group	Population size in multi-species co-evolution
P_{mutate}	Mutation probability in genetic operators

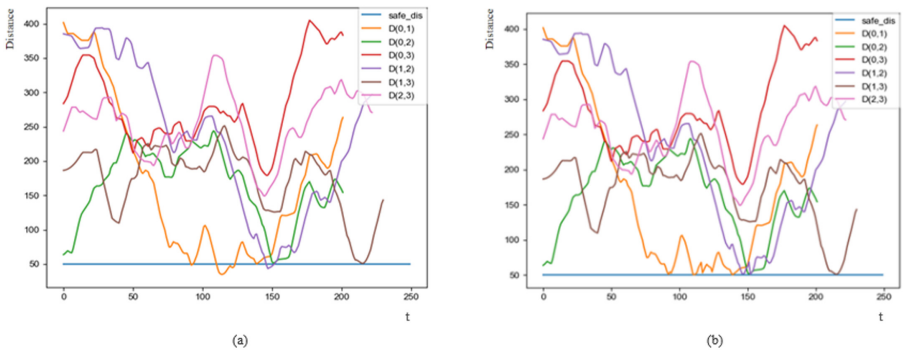


Fig. 5. Path planning results of 4 robots on the tsp225. (a) Robot spacing before optimization. (b) The distance between the robots after optimization.

Algorithm 3: Multi-robot path planning algorithm flow

1 The related parameters of initial adpK-means clustering and GA-DMFOA are described in detail in Table 1
2 Use adpK-means clustering algorithm for task allocation
3 Each data set after allocation is planned by a robot, and the GA-DMFOA algorithm is used for each robot to find the optimal path
4 Use collision detection algorithm 6 to optimize the planned path
5 **if** Collision **then**
6 Adopt a delayed wait strategy
7 **end if**
8 Output the optimal solution for path planning of all robots

4 Experiments

4.1 Parameter Analysis Experiment

A key parameter of the adpK-means clustering algorithm is the scaling factor *Scale*, which is used to dynamically adjust the clustering space. This parameter directly affects the degree of change of the path points in the clustering space in each iteration and has a certain impact on the results of the algorithm. So this article debugs this parameter and tests it in pr76.tsp, kroA100.tsp and tsp225. The value set of the parameter *Scale* is $\{1.05, 1.10.1.15, \cdots, 1.5\}$, for these 10 different values conduct experiment. The number of robots is set to $K = 4$, and the maximum number of iterations is $Kmeans\_\max\_iter = 100$. In order to ensure the accuracy of the experiment, this paper runs the algorithm 20 times independently. It takes the average value of the maximum path length *maxspan* estimated by the greedy strategy as the final evaluation result.

Figure 6(a) is the data distribution diagram of pr76, and the experimental results of the *Scale* parameter are shown in Fig. 6(b). Among them, the resulting curve obtained by K-means clustering only once is marked as K-means, and the result obtained by the adpK-means clustering algorithm is marked as adpK-means. It can be seen from Fig. 6(b) that the experimental results of adpK-means are better than those of K-means under ten sets of *Scale* parameter values, which can prove that the process of the adaptive clustering algorithm is effective. Let's analyze its experimental curve again. As the *Scale* parameter increases, *maxspan* also tends to increase overall. This phenomenon is because the more extensive the *Scale*, the greater the degree of displacement of the points in the clustering space. The clustering results will also experience severe jitter. From the experimental results log in the middle, we also observed that the larger the *Scale*, the larger the clustering results will be in the iterative process. The *maxspan* fluctuates violently into a broader data range. From the experimental results of pr76, the *maxspan* obtained when *Scale* is 1.15 is the minimum value.

The parameter verification on a single data set is difficult to be convincing. This paper conducted the same experiment on the larger-scale data set kroA100 and the ultra-large-scale data set tsp225. The experimental results are shown in Fig. 7(b) and

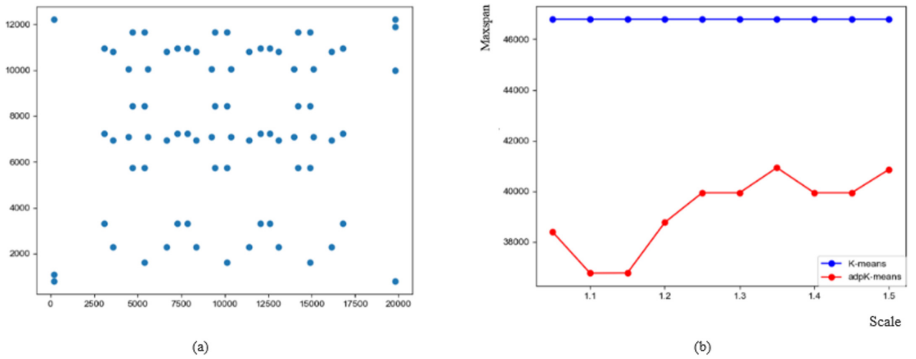


Fig. 6. The experimental results of the scaling factor Scale in the pr76 dataset. (a) pr76 data distribution. (b) pr76 AdpK-means clustering allocation result

Fig. 8(b). In the experimental results, it can still be seen that after the *Scale* gradually increases, the overall *maxspan* is also increasing. However, in the experimental results of these two large-scale data sets, it can be seen that the larger the data scale, the greater the difference in results caused by the changes in *Scale*. Because adpK-means is an algorithm that dynamically adjusts the clustering space, when the data scale is large and the point sets are denser, the *Scale* is too large, which will cause a large number of sample categories to shift, and the results become of difficult to control. The iterative process in adpK-means is to adjust the categories of samples at the edge of each point concentration, and the category of a small number of samples should be changed during each adjustment. Only in this way can we find a better solution, and the algorithm will converge faster. In kroA100 and tsp225, $Scale = 1.1$ obtained the optimal solution in the experiment. From the experimental results of the three data sets, it can be seen that $Scale = \{1.05, 1.1, 1.15\}$ is the three best solutions in the experimental results. Based on this result, this paper uses $Scale = 1.1$ as a benchmark value, and dynamically adjusts it on this basis when experimenting on different datasets.

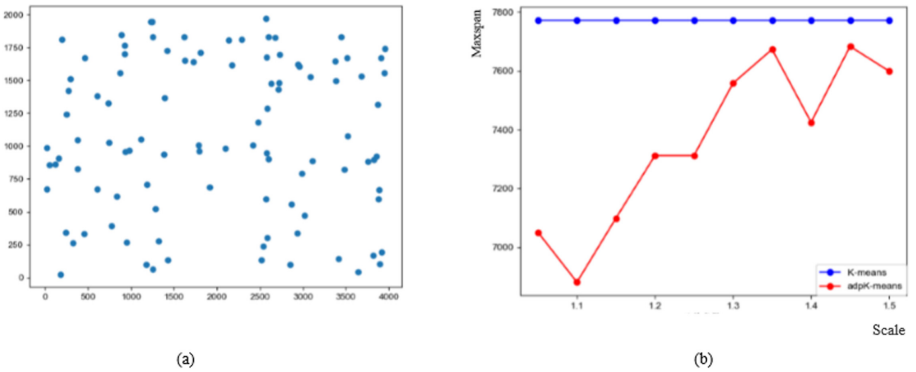
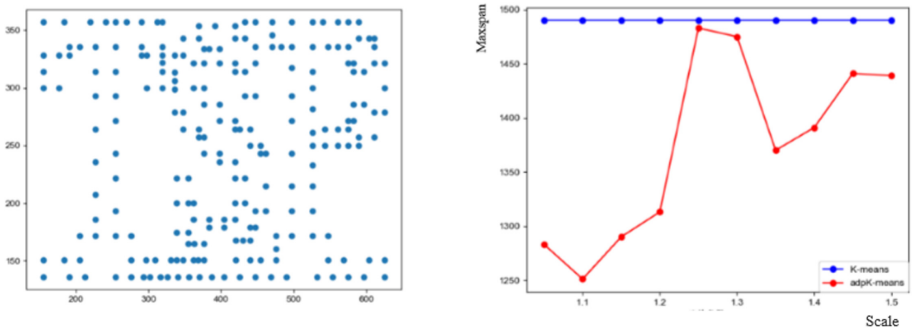


Fig. 7. The experimental results of the scaling factor Scale in the kroA100 dataset. (a) kroA100 data distribution. (b) kroA100 AdpK-means clustering allocation result



(a) (b)

Fig. 8. The experimental results of the scaling factor Scale in the tsp225 dataset. (a) tsp225 data distribution. (b) tsp225 AdpK-means clustering allocation result

4.2 Comparison with Basic K-Means Experiment

In order to verify the effectiveness of the algorithm, this paper compares the adpK-means clustering algorithm (adpK-means) with the basic K-means clustering algorithm (K-means) on different data sets. The path planning part adopts GA-DMFOA algorithm, and the control parameters are consistent. Set the number of robots to $K = 4$, the adaptive clustering scaling factor is $Scale = 1.05$, the number of adaptive clustering iterations is $Kmeans\_\max\_iter = 100$, the maximum number of iterations in the GA-DMFOA algorithm is $MaxIter = 500$, and there are fruit flies in the population The number of individuals is $K = 60$, the number of multi-group cooperation is $Group = 3$, and the probability of mutation in the genetic operator is $Pmuate = 0.05$. In order to avoid accidental interference, the algorithm was independently run 20 times in the experiment, and the optimal value, average value, and worst value of the 20 results were compared. The experimental results are shown in Table 2. From the experimental results, it can be seen that on all the tested data sets, the experimental results obtained by the adpK-means algorithm are better than the K-means regardless of the optimal value, average value, or worst value. The development of the algorithm shows that the effectiveness and applicability of the adpK-means algorithm are very high.

This paper visualizes the running results of the tsp225 data set and analyzes and discusses the intermediate results of its running. Figure 9(a) shows the effect of path allocation after using K-means clustering. In the figure, four different colors are used to represent the results of varying robot allocation, and the path obtained under the greedy strategy is drawn. Figure 9(b) shows the visualization of adpK-means results. It can be seen from the figure that the clustering results of adpK-means and K-means are inconsistent. According to the calculation process of the adaptive clustering algorithm, it can be seen that the adpK-means clustering has found a better solution in the iterative process. Observing the path curve drawn in the figure, it can be seen that the path is still staggered, which further proves that the path planned under the greedy strategy is not the optimal solution, and the GA-DMFOA algorithm is needed to prepare the path further.

Table 2. Comparison of experimental results of adpK-means clustering algorithm.

Dataset	Method		Optimal	Average	Worst
St70	GA-MDFOA	+K-means	199.87	200.10	228.56
		Adp K-means	**197.56**	**199.28**	**223.67**
rat99	GA-MDFOA	+K-means	376.57	388.10	414.36
		Adp K-means	**340.41**	**363.08**	**366.78**
ch150	GA-MDFOA	+K-means	2070.66	2082.37	2131.75
		Adp K-means	**2009.63**	**2056.08**	**2122.88**
tsp225	GA-MDFOA	+K-means	1237.29	1270.57	1298.11
		Adp K-means	**1180.17**	**1228.48**	**1263.70**
tsp442	GA-MDFOA	+K-means	16024.63	16362.57	16716.76
		Adp K-means	**15988.97**	**16316.12**	**16716.27**

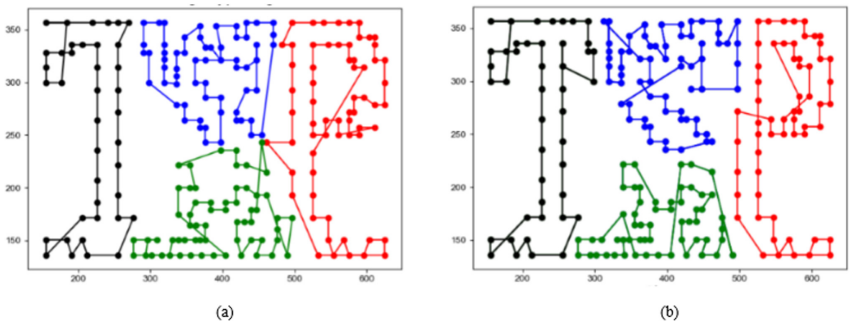


Fig. 9. On the tsp225 dataset, the path initialized after apdK-means is allocated. (a) K-means. (b) apdK-means.

Table 3 shows the path length of each robot according to the greedy strategy. The robot corresponds to the four colors of red, green, blue and black in Fig. 9(a) according to the serial number. Directly from the maximum path length *maxspan*, adpK-means obtains better results. Observing the data of each robot in the table, we can see that the K-means distribution results are not balanced. The maximum and minimum values are 1358.25 and 965.48, respectively, while the maximum and minimum values are 1188.67 and 1047.10 in the results of adpK-means. It can be seen that adpK-means is deployed between different robots, which makes the workload of each robot more balanced.

Table 3. The path length of each robot in the clustering result of tsp225.

Method	Maxspan	Robot1	Robot2	Robot3	Robot4
K-means	1358.25	1358.25	1182.47	**1042.91**	**965.48**
AdpK-means	**1188.67**	**1188.67**	**1092.34**	1047.10	1161.31

In the second stage, the path planning needs to be redone after the path point allocation. Table 4 shows the results of GA-DMFOA path planning. It can be seen from the results that compared to Table 3, *maxspan* has been optimized slightly, which proves the effectiveness of GA-DMFOA in the path planning stage, and also reflects that the path obtained by the greedy strategy is closer to the optimal solution. Therefore, it is reasonable to use the path length obtained by the greedy strategy as the workload of the robot. The path planning results after K-means and adpK-means are allocated shown in Fig. 10(a) and Fig. 10(b).

Table 4. Comparison of results after path planning.

Method	Maxspan	Robot1	Robot2	Robot3	Robot4
K-means	1237.29	1237.29	1049.92	966.37	**920.74**
AdpK-means	**1180.17**	**1154.30**	**1016.58**	**946.59**	1180.17

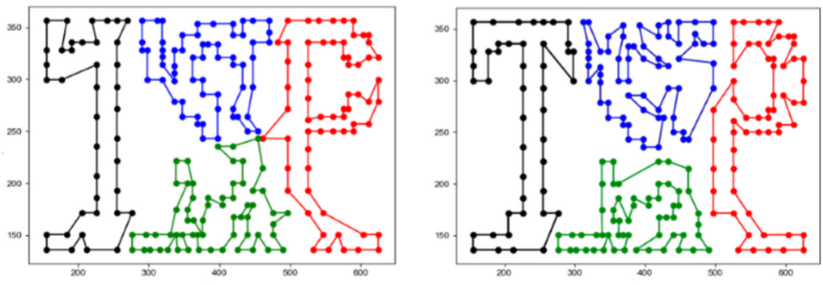


Fig. 10. On the tsp225 dataset, the effect of different allocation strategies on path planning. (a) K-means. (b) apdK-means.

5 Conclusions and Future Work

Multi-robot systems are becoming an intensively investigated area of modern robotics in the last few years. The key to using multi-robot systems is coordination. This paper proposes an adaptive K-means clustering algorithm. Through experimental verification, the algorithm improves the balance of multi-robot system task allocation, and reduces the overall consumption of the multi-robot system, and improves the traditional K-means algorithm from being initialized. And the influence of outliers. In the future, it is necessary to consider how to allocate tasks when there are multiple tasks of different types in a multi-robot system, that is, the task allocation of a heterogeneous multi-robot system.

References

1. Ahmadi, M., Stone, P.: A multi-robot system for continuous area sweeping tasks. In: Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006, pp. 1724–1729. IEEE (2006)
2. Kumar, V., Michael, N.: Opportunities and challenges with autonomous micro aerial vehicles. Int. J. Robot. Res. **31**(11), 1279–1291 (2012)
3. Freund, E.: Fast nonlinear control with arbitrary pole-placement for industrial robots and manipulators. Int. J. Robot. Res. **1**(1), 65–78 (1982)
4. Burgard, W., Moors, M., Stachniss, C., et al.: Coordinated multi-robot exploration. IEEE Trans. Rob. **21**(3), 376–386 (2005)
5. Fong, T., Thorpe, C., Baur, C.: Multi-robot remote driving with collaborative control. IEEE Trans. Industr. Electron. **50**(4), 699–704 (2003)
6. Gerkey, B.P., Matarić, M.J.: A formal analysis and taxonomy of task allocation in multi-robot systems. Int. J. Robot. Res. **23**(9), 939–954 (2004)
7. Bektas, T.: The multiple traveling salesman problem: an overview of formulations and solution procedures. Omega **34**(3), 209–219 (2006)
8. Wagner, G., Choset, H.M.: A complete multirobot path planning algorithm with performance bounds. In: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3260–3267. IEEE (2011)
9. Berhault, M., Huang, H., Keskinocak, P., et al.: Robot exploration with combinatorial auctions. In: Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No. 03CH37453), vol. 2, pp. 1957–1962. IEEE (2003)
10. Chu, P.C., Beasley, J.E.: A genetic algorithm for the generalised assignment problem. Comput. Oper. Res. **24**(1), 17–23 (1997)
11. Akkiraju, R., Keskinocak, P., Murthy, S., et al.: An agent-based approach for scheduling multiple machines. Appl. Intell. **14**(2), 135–144 (2001)
12. Starke, J., Schanz, M., Haken, H.: Self-organized behaviour of distributed autonomous mobile robotic systems by pattern formation principles. In: Distributed Autonomous Robotic Systems, vol. 3, pp. 89–100. Springer, Berlin, Heidelberg (1998)
13. Kwok, K., Driessen, B.J., Phillips, C.A., et al.: Analyzing the multiple-target-multiple-agent scenario using optimal assignment algorithms. Journal of Intelligent and Robotic Systems (2002)
14. Xu, X., Yuan, H., Liptrott, M., et al.: Two phase heuristic algorithm for the multiple-travelling salesman problem. Soft. Comput. **22**(19), 6567–6581 (2018)
15. Xie, P.: Research on Welding Path Planning of Automobile Interior and Exterior Parts Based on Multi-Robot Collaboration. South China University of Technology, Guangdong (2018)
16. Shi, D.: Research on the Path Planning of the Body-in-White Welding Robot Based on the Secondary Development of ROBCAD. Hefei University of Technology, Hefei (2014). In Chinese
17. Burger, M., Su, Z., De Schutter, B.: A node current-based 2-index formulation for the fixed-destination multi-depot travelling salesman problem. Eur. J. Oper. Res. **265**(2), 463–477 (2018)
18. Kitjacharoenchai, P., Ventresca, M., Moshref-Javadi, M., et al.: Multiple traveling salesman problem with drones: mathematical model and heuristic approach. Comput. Ind. Eng. **129**, 14–30 (2019)
19. Qu, H., Yi, Z., Tang, H.J.: A columnar competitive model for solving multi-traveling salesman problem. Chaos, Solitons Fractals **31**(4), 1009–1019 (2007)
20. Kanungo, T., Mount, D.M., Netanyahu, N.S., et al.: An efficient k-means clustering algorithm: analysis and implementation. IEEE Trans. Pattern Anal. Mach. Intell. **24**(7), 881–892 (2002)

21. Hartigan, J.A., Wong, M.A.: Algorithm AS 136: a k-means clustering algorithm. J. Roy. Stat. Soc. Ser. c (Appl. Stat.) **28**(1), 100–108 (1979)
22. Loohach, R., Garg, K.: Effect of distance functions on k-means clustering algorithm. Int. J. Comput. Appl. **49**(6), 7–9 (2012)
23. Sinwar, D., Kaushik, R.: Study of euclidean and manhattan distance metrics using simple k-means clustering. Int. J. Res. Appl. Sci. Eng. Technol. **2**(5), 270–274 (2014)
24. Sahu, L., Mohan, B.R.: An improved K-means algorithm using modified cosine distance measure for document clustering using Mahout with Hadoop. In: 2014 9th International Conference on Industrial and Information Systems (ICIIS), pp. 1–5. IEEE (2014)
25. Celebi, M.E., Kingravi, H.A., Vela, P.A.: A comparative study of efficient initialization methods for the k-means clustering algorithm. Expert Syst. Appl. **40**(1), 200–210 (2013)
26. Kivelevitch, E., Cohen, K., Kumar, M.: A market-based solution to the multiple traveling salesmen problem. J. Intell. Robot. Syst. **72**(1), 21–40 (2013)
27. Arthur, D., Vassilvitskii, S.: k-means++: the advantages of careful seeding. Stanford (2006)
28. Reinelt, G.: TSPLIB—a traveling salesman problem library. ORSA J. Comput. **3**(4), 376–384 (1991)
29. Wu, L., Zuo, C., Zhang, H.: A cloud model based fruit fly optimization algorithm. Knowl.-Based Syst. **89**, 603–617 (2015)
30. Mousavi, S.M., Alikar, N., Niaki, S.T.A., et al.: Optimizing a location allocation-inventory problem in a two-echelon supply chain network: a modified fruit fly optimization algorithm. Comput. Ind. Eng. **87**, 543–560 (2015)